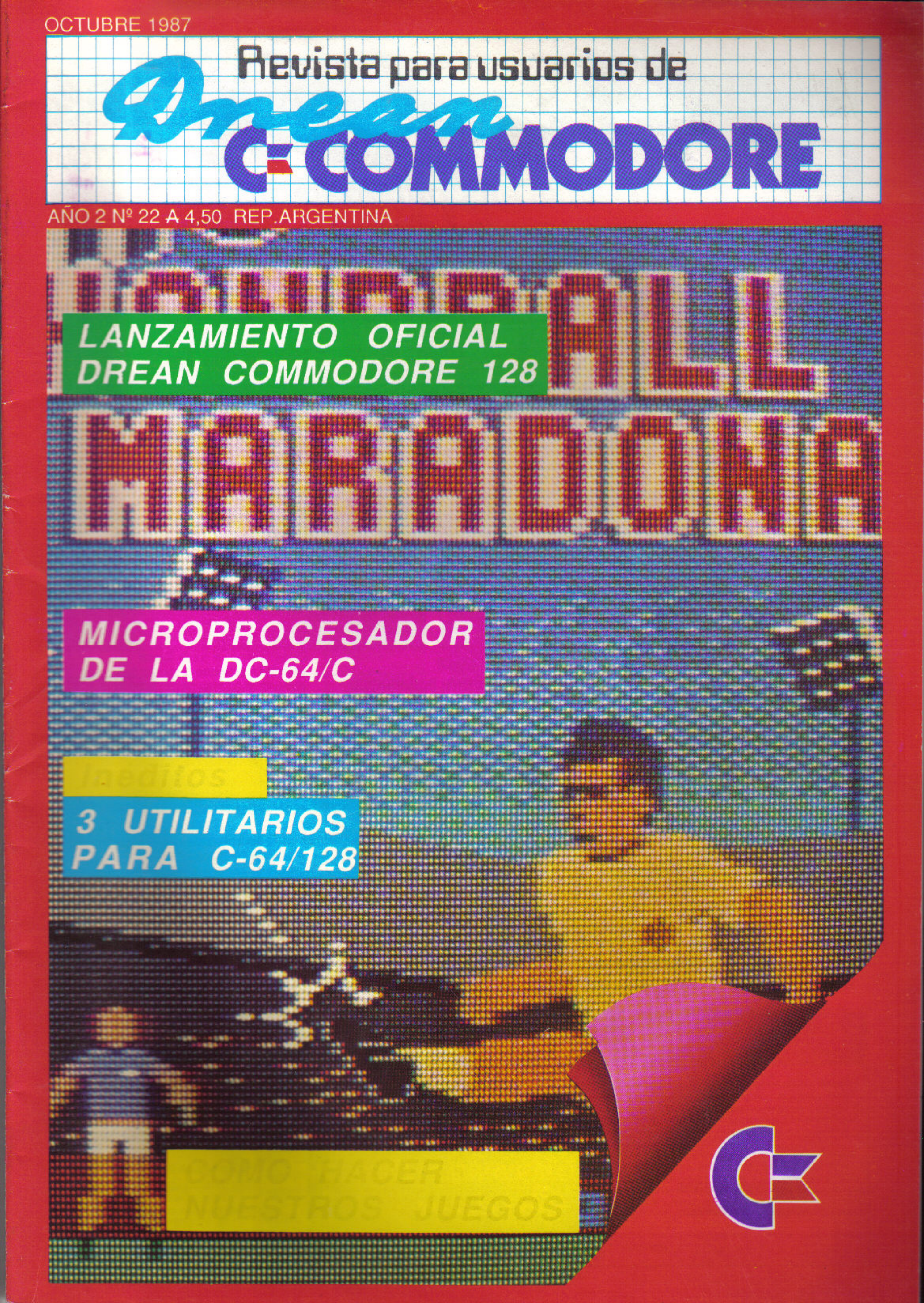


OCTUBRE 1987
Revista para usuarios de
Drean
COMMODORE
AÑO 2 Nº 22 A 4,50 REP.ARGENTINA
LANZAMIENTO OFICIAL
DREAN COMMODORE 128
MARADONA
MICROPROCESADOR
DE LA DC-64/C
Inéditos
3 UTILITARIOS
PARA C-64/128
COMO HACER
NUESTROS JUEGOS

SUMARIO

NOTAS TECNICAS

PROGRAMAS

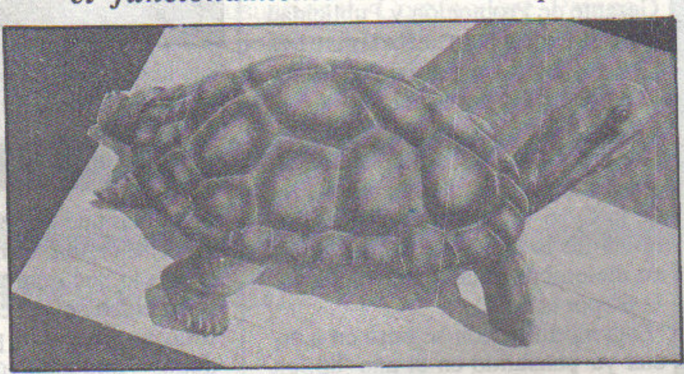
REVISION DE SOFT

NOTICIAS

SECCIONES FIJAS



Indagamos las características del microprocesador de la Drean Commodore 64/C y les explicamos porque hace posible el funcionamiento de esta computadora.



El paso de BASIC a PASCAL y después a C es uno de los factores que inducen a la programación en serio, y a la toma de conciencia del potencial de una computadora.



Cuando se compra una computadora con deseos de programar usualmente se está pensando en la realización del juego propio. Aprovechando las posibilidades que brinda el BASIC 7.0 les contamos cómo hacerlos.

Drean Commodore AÑO 2 Nº 22 1987 4,50

Director General
Ernesto del Castillo

Director Editorial
Cristián Pusso

Director Periodístico
Fernando Flores

Secretario de Redacción
Ariel Testori

Arte y Diagramación
Fernando Amengual
Tamara Migelson

Fotografía
Victor Grubicy
Image Bank
Eduardo Comesaña

Departamento de Avisos
Oscar Devoto
Nelso Capello

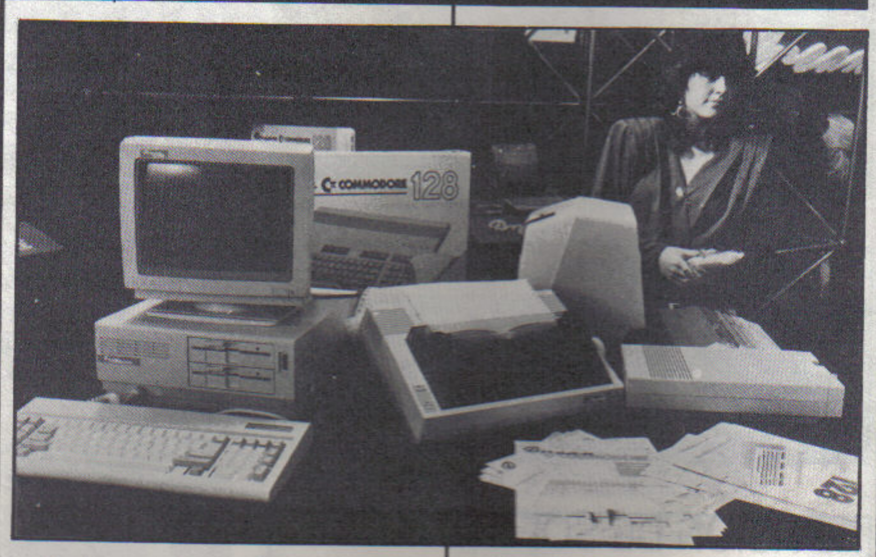
Revista para usuarios de Drean Commodore es una publicación mensual editada por editorial **PROEDI S.A.**, Paraná 720, 5º Piso(1017) Buenos Aires. Tel.: 46-2886 y 49-7130. Reg. Nac. de la Prop. Intelectual E.T., M. Registrada. Precio de este ejemplar, ₳ 4,50. Los ejemplares atrasados se venderán al precio del último número en circulación. Queda hecho el depósito que indica la Ley 11.723 de Propiedad Intelectual ISSNO326-8233

Impresión: Calcotam. **Fotocromo tapa**: Columbia.

Distribuido en Capital: Martino, Juan de Garay 358, P.B. Capital. **Distribuidor interior**: DGP, Hipólito Yrigoyen 1450, Capital Federal. T.E.: 38-9266/9800.

LANZAN LA DREAN COMMODORE 128

Con la "presentación en sociedad" de la C-128 en septiembre, el próximo lanzamiento de las líneas de PC compatibles con IBM, el refuerzo de la presencia en el sector de informática educativa y la expansión de los **Centros de Atención al Usuario** y los servicios de asistencia técnica, **Drean-Commodore** da "una nueva muestra de continuidad en su línea de trabajo" aseguró el Gerente de Promoción y Publicidad de esa empresa, **Felipe Mc Gough,** al lanzarse oficialmente en la Argentina la popular 128.



Los trabajos de la empresa en el área informática -recordó Mc Gough- comenzaron con el lanzamiento, el de la C-16 y, en forma casi simultánea de la 64. Posteriormente en diciembre pasado se reemplazó la 64 por la 64-C.

"Esta modificación se puso en marcha ya pensando en el año '88, en cuyo transcurso se terminaría de armar toda una línea informática y que ahora se delinea con la presentación oficial de la C-128, en septiembre", enfatizó.



Felipe Mc Gough,

Junto con este lanzamiento, se presentó el 64 utilitario, también conocido como el **PC-compacto,** que viene con un gabinete, disquetera incorporada, monitor monocromático e impresora, es decir un conjunto completo, además del soft básico.

En forma simultánea se presenta todo el proyecto de informática educativa y su plan pedagógico.

"Las estadísticas -relató al justificar la importancia de esta iniciativa- nos demuestran que la mayoría de las máquinas que hay en las escuelas de Argentina son Commodore, y nosotros lo que queremos es darles la posibilidad de que ellos trabajen con todo un proyecto pedagógico por detrás", que incluye obviamente un gran volumen de soft especialmente diseñado.

Mc Gough explicó que para el lanzamiento de la 128 fue necesario efectuar una serie de trabajos en la planta que la empresa tiene en San Luis, para adaptarla a la nueva línea de producción.

Añadió que la mayor cantidad de componentes de cada máquina son de producción local, pero sin embargo los"cerebros", es decir los chips, se importan, dado que se producen en el país.

Asimismo reveló que, respecto de la "vida útil" de la C-64, en Estados Unidos se calcula que esta máquina va a tener una participación activa en el mercado hasta el año 1990 o 1991, período que en Argentina podría extenderse un par de años más debido a que aquí llegó más tarde.

Respecto del rendimiento de las computadoras Drean-Commodore, Mc Gough admitió que se detectaron algunos problemas en cuanto a la norma. "Hay mucho soft en el mercado, de origen importado que fue desarrollado para el sistema de video NTSC, y debido a la modificación a Pal-N, algunos programas no se cargaban. Esto nos representó muchos dolores de cabeza y tareas de desarrollo por parte nuestra y de Commodore en Estados Unidos".

"Por ese motivo -añadió- la C-128 sale en sistema NTSC, porque se trata de otro mercado y el usuario busca evitar problemas como, por ejemplo, que en determinado momento la máquina no cargue un soft desarrollado especialmente para médicos, control de stock o algún otro uso profesional".

Acerca de los **Centros de Asistencia al Usuario** dijo que se reveló que se realizó un estudio que aconsejó el traspaso en algunos casos de los titulares de los servicios técnicos, cuando no respondían a las necesidades del mercado.

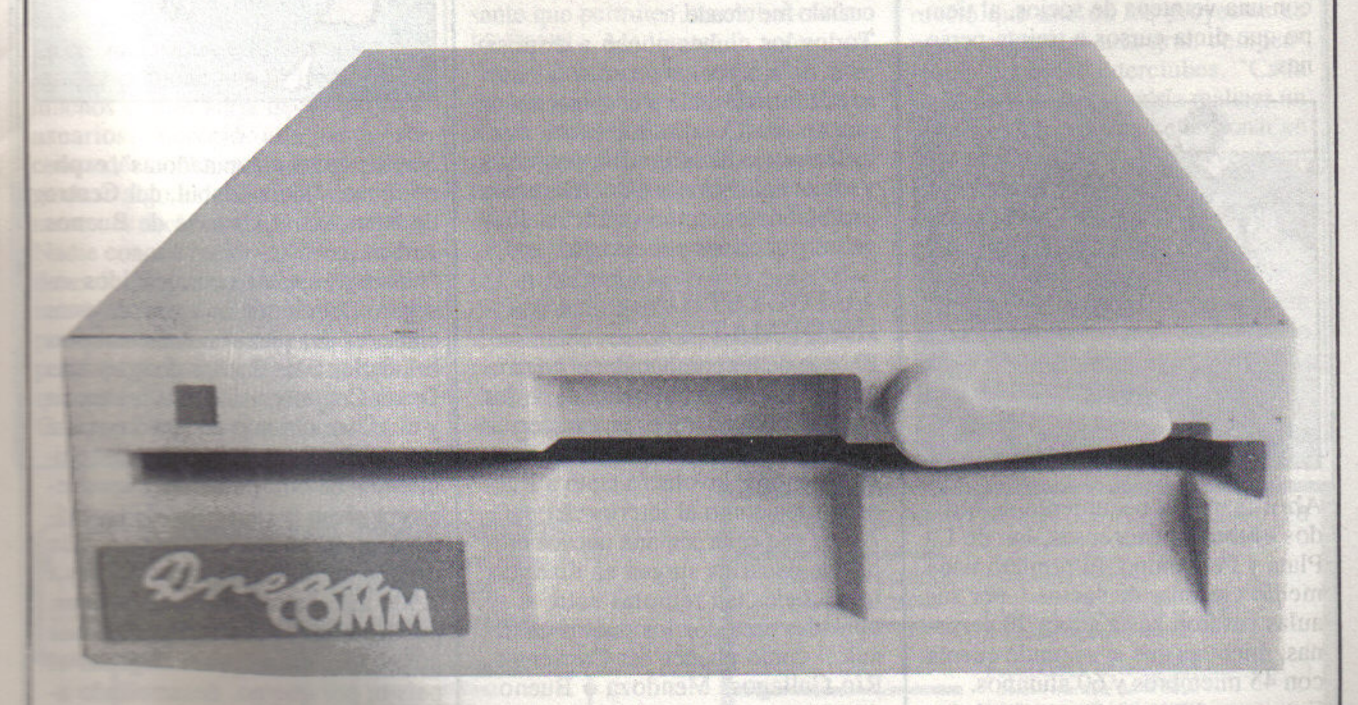
En este campo se puso también en marcha en los diferentes Centros de Asistencia al Usuario, y como posibilidad opcional para cada nuevo poseedor de una Drean-C ommodore, la instalación a domicilio de las máquinas, algo que Mc Gough calificó como "una manera de romper el hielo".

Respecto de la PC compatible con IBM, adelantó que la presentación "en sociedad" se efectuará alrededor de noviembre o diciembre y aseguró que estas máquinas tendrán un precio "muy accesible".

CONCURSO de NOTAS, TRUCOS y PROGRAMAS

Los trabajos pueden estar referidos tanto a la Drean-Commodore 64 como a la 128.
Las notas deben apuntar a un mejor aprovechamiento de las potencialidades de las máquinas.
Los trucos y los programas tienen que ser inéditos.
El software deben remitirlo a nombre de Concurso Revista para usuarios de Commodore, Paraná 720, 5º piso, (1017) Capital Federal, grabado en disquete o casete. Indiquen los datos del programa, computadora y autor. Es preferible (pero no imprescindible) que acompañen el listado del mismo por impresora.

un DRIVE Drean-Comm.



Menciones: 10 cartridges

CIERRE DEL PRIMER CERTAMEN: 27 de noviembre de 1987.

MAS CLUBES EN EL INTERIOR

La explosión de los clubes para usuarios para la familia "commodoriana" se produce en todo el país. Para explicar el porqué y el cómo del fenómeno entrevistamos a Carlos Dabul, nexo entre el Centro de Buenos Aires y los del interior.



La familia "commodoriana" sigue creciendo en todo el país; y la mejor prueba es que en los últimos meses gran cantidad de clubes surgieron a lo largo del país y sólo cinco de ellos ya cuentan con más de dos centenares y medio de socios y dictan cursos a casi 400 personas.

Tandil, Pergamino, La Plata, Mendoza y Corrientes son los ejemplos que citamos ahora. El de Tandil fue creado en marzo último, y cuenta con una veintena de socios, al tiempo que dicta cursos a treinta personas.



Abril marcó el nacimiento de otros dos clubes bonaerenses, los de La Plata y Pergamino. El primero tiene medio centenar de socios y por sus aulas pasaron hasta ahora 70 personas, mientras que el segundo cuenta con 45 miembros y 60 alumnos.

Entretanto en mayo fueron fundados los clubes de usuarios de Mendoza y Corrientes. El primero tiene 45 socios y 80 alumnos, y el correntino dispone de 120 miembros y casi un centenar y medio de inscriptos en sus diferentes cursos.

Esta "explosión" de los clubes para reunir a los felices poseedores de computadoras hogareñas Drean-Commodore también incluye, por ejemplo, a Santa Fe y Río Gallegos. En la ciudad litoraleña -donde el club funciona desde hace un año- sus 70 socios intercambian programas, experiencias y asisten a los diferentes cursos, mientras en la urbe santacruceña los 25 socios desarrollan una intensa actividad que llevó, incluso, al dictado de cursos a dos centenares de personas desde diciembre pasado, cuando fue creado.

Todos los clubes ponen a disposición de sus socios no sólo los cursos de introducción a la operación de una Drean-Commodore, sino también manejo de utilitarios, lenguajes y los de aplicaciones específicas para profesionales, como médicos, abogados y docentes por ejemplo.

AUTOGESTION INFORMATICA

El auge de las computadoras hogareñas, y muy especialmente el de las Drean-Commodore, es una tendencia que, desde su mismo lanzamiento en la Argentina, involucró tanto a Buenos Aires como al interior del país. Así es que cada semana nuevos clubes de usuarios surgen en distintas localidades tan remotas entre sí -y ubicadas en todos los puntos cardinales- como pueden ser Corrientes, Río Gallegos, Mendoza o Buenos Aires.

Los clubes son el ámbito natural donde los miembros de la gran familia "commodoriana" se reúnen "para intercambiar experiencias, software y bibliografía, asistir a cursos o participar en diferentes certámenes, en los que pueden demostrar sus aptitudes y progresos en el empleo de las computadoras", explicó Carlos Daniel Dabul, del Centro de Atención al Usuario de Buenos Aires.

"Nosotros estamos convencidos -aseguró Dabul- que cada uno de estos clubes es una pieza fundamental para brindarles a los dueños de máquinas Drean-Commodore toda la asistencia y el asesoramiento necesarios, para que puedan progresar en sus conocimientos de computación y aprovechar a pleno la capacidad de las máquinas; en síntesis, para que, aún estando quizás en plena cordillera y a 2.500 kilómetros de Buenos Aires, puedan estar al tanto de las últimas novedades en software llegadas al país o, por qué no, desarrollado aquí".

- ¿Hay muchos clubes formados ya en todo el país?.

- Sí, son una veintena, cifra que a nuestro juicio es bastante importante, pero no todo lo que quisiéramos- admitió Dabul.

- ¿Eso quiere decir que van a alentar el surgimiento de

nuevas sedes?
-¡Por supuesto!- enfatizó, para luego revelar que "nuestro propósito es que muchos de los clubes surjan de manera espontánea; que sean los propios usuarios del interior del país los que tomen la iniciativa, se reúnan con sus vecinos que ya disponen también de computadoras Drean-Commodore y avancen con nuestra ayuda en la organización de sus centros".
Consultado acerca de las complejidades de ese proyecto, Dabul hizo hincapié en que "ya tenemos una experiencia importante sobre los proglemas que pueden aparecer, y los pasos necesarios para comenzar a trabajar, así que estamos listos para darles asesoramiento a nuestros amigos del interior que quieran acometer esta tarea".
"Queremos integrarnos. Hay mucha gente capaz en todo el país que puede aportar gran cantidad de elementos al desarrollo de la informática. Si trabajamos en una suerte de gran equipo, si intercambiamos las experiencias entre todos nosotros, podemos aprender mucho en conjunto", destacó.
La pregunta acerca de las causas por las que propone que la inicativa de muchos clubes surja de los propios usuarios, mereció una respuesta contundente: "La mejor forma de organizar algo es que crezca naturalmente, es decir de abajo hacia arriba. Nadie conoce mejor las necesidades de un lugar que sus propios habitantes. Nosotros podemos aportar experiencas, antecedentes, encuadres; pero ninguno sabe mejor que es lo que necesitan los usuarios de Drean-Commodore, por ejemplo, en Tartagal o El Calafate, que los propios habitantes de esas regiones".


Carlos Daniel Dabul

- ¿Y las aplicaciones educativas?.
-Esa es otra perspectiva muy interesante que permiten los clubes- señaló Dabul-. Mediante el apoyo de Drean-Commodore y cada club, podemos colaborar con las escuelas, los docentes y los miembros de las cooperadoras para instalar gabinetes educativos en los establecimientos escolares. El papel primordial de los clubes y Drean-Commodore pasaría por el asesoramiento y la provisión del soft educativo.
Consultado acerca del costo de poner en marcha un Club de Usuarios, Dabul relativizó el problema. "Todo depende de cuántos sean los que trabajan con Drean-Commodore en una ciudad. Obviamente si hay 200 interesados, habrá que pensar en más espacio, en mayores gastos; pero también son más personas para solventarlo. Si en cambio hablamos de un grupo de 10 ó 20, todo se reduce en forma proporcional".
"Además, como señalé recién -recordó- el club le facilitaría a sus socios las últimas novedades en software, repuestos, bibliografía y accesorios; esto significa mantener un vínculo constante con todo lo que se está haciendo en cualquier punto del país sin contar que, en la medida que se avance y progrese, cada uno de ellos podría comenzar eventualmente a brindar distintos cursos y, por qué no, llegado el caso, servir como nexo con Drean para el service y mantenimiento de cada computadora".
Otro de los objetivos es la creación de un carnet con alcance nacional, para que un socio de cualquier club del país pueda recibir asesoramiento y los mismos beneficios que en su ciudad de origen, cuando está de visita en otro sitio.
Entre las ideas que maneja, Dabul reveló que uno de los proyectos es poder organizar una gran compentencia nacional interclubes. "Cada club, por ejemplo, podría realizar un concurso interno para seleccionar su representante y luego intervenir en un certamen de alcance nacional que premie tanto al usuario, como al club y, por ende, a todos los socios de la localidad a la que pertenece".
"Esto -aseguró Dabul como colofón de la charla- es apenas un botón de muestra. Todos juntos podemos hacer grandes cosas si nos lo proponemos".

EL MICROPROCESADOR DE LA 64 (1º PARTE)

Indagaremos las características de este componente que hace posible el funcionamiento de una computadora.

¿Qué es, ante todo, un microprocesador? Físicamente -en el caso de 6510- es una pastillita negra de 5 cm. de largo por 1 cm. de ancho, con 40 patitas y de aspecto inofensivo.

Pero dentro de ese encapsulado existe una gran cantidad de compuertas lógicas, registros y buffers cuyo sistemático y ordenado trabajo permite la ejecución de los programas.

Buscando analogías podemos figurarnos al microprocesador como un estricto y meticuloso director de orquestà que dirige según una partitura (el programa) a una vasta cantidad de instrumentos, que en nuestro caso serían los distintos registros y posiciones de memoria que rodean al microprocesador y que forman parte de la computadora.

A través del tiempo y desde la creación de los legendarios microprocesadores de 4 bits, a estos directores de orquesta electrónicos se los ha ido desarrollando sin pausa, mejorando su capacidad, su estructura interna, su versatilidad y, por sobre todas las cosas, su velocidad.

Hoy, sin embargo, nos detenemos a observar un microprocesador que podría resultar modesto, frente a los desarrollados actualmente, pero que tiene una importancia capital para quienes desean saber quién es y cómo trabaja el corazón que late y anima a nuestra Drean-Commodore.
Señoras y señores: les presentamos al microprocesador 6510.

EL ARBOL GENEALOGICO

El 6510 es una elaboración del famoso 6502 que la no menos famosa computadora APPLE II lleva en su interior.
La popularidad y difusión de este chip en aquel momento alentó a la



casa Commodore a incluirlo en su legendaria comptuadroa VIC-20.
Actualmente Commodore sigue utilizándolo, pero como controlador inteligente en las unidades de disco 1541 y 1571.
Entre el 6502 y el 6510 encontramos algunas diferencias.
Desde el punto de vista de conexionado cambian la distribución de las patitas aunque se mantiene la cantidad. Pero en el orden constructivo, el 6510 incluye un registro bidireccional, que el 6502 no tiene, el cual le permite intercambiar datos con otros dispositivos.

No obstante el set de instrucciones de ambos es idéntico, lo que permite una total adaptación entre microprocesadores al programar en lenguaje de bajo nivel.

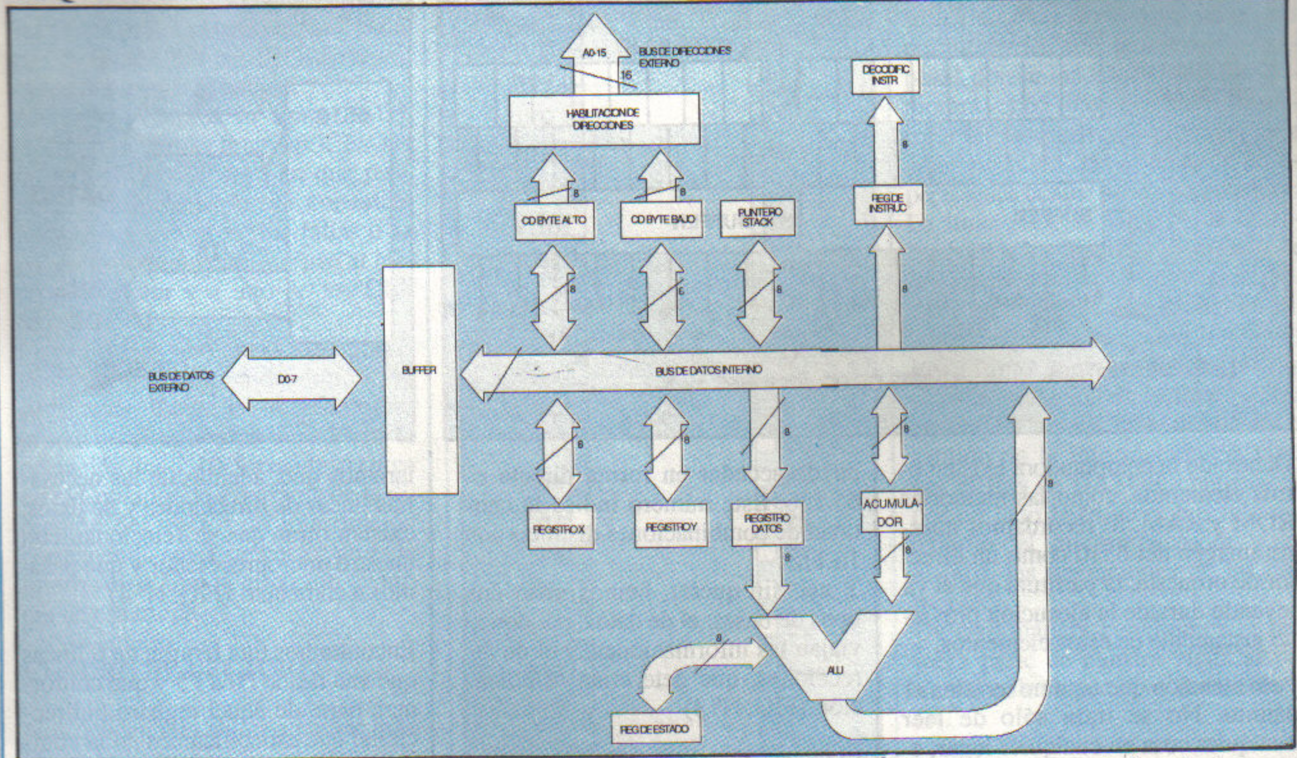
Aclaramos que el lenguaje de bajo nivel es el idioma que el micro comprende directamente sin necesidad de intermediarios.
Este lenguaje está formado por cifras binarias de 1 byte de longitud y dentro del cual encontramos dos categorías de palabras: las que son instrucciones y las que no lo son.

Justamente, las instrucciones son las que determinan la tarea a realizar por el microprocesador, y la cantidad de instrucciones que este admite habla de la variedad de tareas que puede realizar.
En el caso del 6510 nos encontramos con 56 instrucciones distintas, pero que son adaptables para uno o más de los 13 modos posibles de direccionamiento, de manera que nos podemos encontrar hasta con 151 códigos diferentes durante su funcionamiento.
Pero la historia no ha terminado. El 6510 nos ha legado un sucesor que hoy se encuentra en el interior de la 128: el 8502.
Este nuevo microprocesador -que mantiene el mismo código de operación de sus antecesores- tiene una importante característica que lo distingue de ellos: poder trabajar al doble de velocidad, hecho que lo habilita para ejecutar un programa en la mitad del tiempo.

ESTRUCTURA INTERNA

Como ya adelantamos, el microprocesador tiene en su interior una serie de contadores y registros que son los artífices de las distintas operaciones que se pueden realizar.
Como es lógico imaginar, estos dispositivos no actúan por sí solos - a su antojo- sino que trabajan armó-

ESQUEMA DEL MICROPORCESADOR 6510



nicamente de acuerdo con las instrucciones y datos que reciben en forma de números.

Una buena pregunta sería: ¿cómo hace el microprocesador para distinguir un número que representa un código de otro que es parte de una dirección? Recordemos que los números que recibe son cifras binarias a las que llamamos bytes.

La respuesta es sencilla: el microprocesador comienza su trabajo suponiendo que el primer número que encuentra es un código de operación. Sobre la base de esta suposición lo toma y lo lleva a uno de sus registros interiores llamado INSTRUCTION REGISTER (Registro de Instrucción) para hacerlo pasar a otro registro denominado INSTRUCTION DECODER (Decodificador de Instrucción) donde el número es descompuesto y analizado.

-FIGURA 1-

Del análisis de este código surge qué tipo de operación se va a realizar y qué modo de direccionamiento se empleará durante su ejecución.

De acuerdo con eso el microprocesador sabrá que una determinada cantidad de números que acompañan al código serán destinados a complementar la operación y por lo tanto NO se tratará de códigos de operación (y no serán decodificados).

Acto seguido la operación es ejecutada.

El encargado de preparar todo lo necesario para efectuar la operación es el Decodificador de Instrucción, que le avisa a un registro que envíe datos y a otro, que los reciba, e instruye al resto para que no intervenga.

Una vez que esa operación finaliza se toma otro número que se considera como un nuevo código de operación. Este será decodificado y así la historia se repetirá... hasta que la computadora se apague.

Esto nos indica que no existe diferencia entre el número 6E -código de operación- y el número 6E que es un complementario de dicha operación, excepto en su ubicación relativa dentro del programa.

Ya podemos imaginarnos que programar en lenguaje de máquina exige mucha atención, porque olvidar un número o invertir su orden de operación puede originar tal desfasaje que se armaría una gran ensalada de códigos.

Y aquí vale la pena señalar una diferencia fundamental entre el hombre y la máquina: la capacidad de admitir ambigüedades.

Si repentinamente sobre un papel con una dirección que dice SAN MARTIN 6345 8vo. "g" cae una gota de tinta que impide ver los dos últimos números de la misma y necesitamos llegar, podemos dirigirnos a una dirección aproximada e iniciar una búsqueda por tanteo en los edificios de la cuadra donde exista un 8vo. "g" hasta localizar el nuestro.

Una computadora en un caso similar no podría hacer absolutamente nada. Ella necesita total precisión en los datos. Sin errores ni ambigüedades. De lo contrario se perderá para siempre en las nubes de Ubeda.

DIRECCIONAMIENTO

Bien, ya sabemos que las instrucciones son decodificadas y ejecutadas, pero ¿de dónde salen esas instrucciones, quién las trajo?

Al comenzar la nota mencionamos que el microprocesador tiene 40 patitas. Estas lo comunican con el resto de los componentes de la computadora.

Algunas de ellas conforman el bus de datos, que está constituido por ocho líneas independientes por donde viajan las informaciones a las que antes nos referíamos: códigos de operación y operandos.

De esta manera los códigos que son el "alimento" del 6510 viajan por estas líneas hasta llegar a sus registros internos y ser digeridos. ¿Pero de dónde salen?.

Figura 2

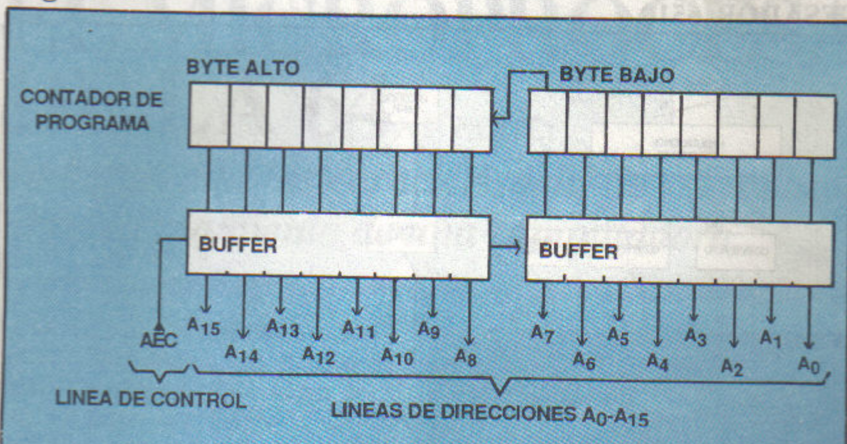
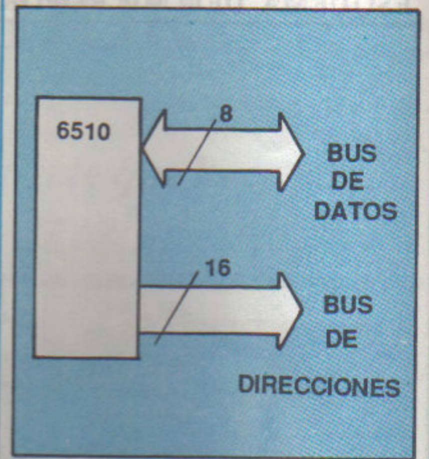


Figura 3



Dentro de la computadora existen otros elementos, como las memorias ROM y RAM. Si recordamos nuestra imagen del 6510 como un director de orquesta, la partitura que él va leyendo durante la ejecución proviene justamente de estos elementos.

Pero atención que aquí no termina el asunto. No se trata sólo de leer cualquier cosa en cualquier lugar, existe un orden para la lectura y alguien tiene que indicarle al microprocesador dónde mirar. Para eso existe otro conjunto de patitas que se encarga de localizar en cada instante aquello que el microprocesador necesita para trabajar.

Estas patitas -que en nuestro caso son 16- se denominan BUS de direcciones. (FIG. Nº 2)

Este bus de direcciones está controlado por un registro llamado Program Counter (contador de programa) y su trabajo se parece al de una telefonista que está discando permanentemente el número de la próxima persona con la que el microprocesador quiere comunicarse.
El trabajo parece terrible pero no lo es; porque en general los números discados son sucesivos (532; 533; 534; etcétera) excepto en caso de que haya saltos (Branch, Jump) donde la telefonista discará un número distinto, pero siempre indicado por el 6510.
Siendo 16 la cantidad de líneas, 16 ha de ser la longitud del Contador de programa y 16 la cantidad de números a discar. Pero a no asustarse, que aquí también los números son binarios y por lo tanto la cantidad total de abonados a los cuales el micro puede acceder en forma directa es 65.536. Este número indica la cantidad de combinaciones distintas con 16 bits.
Y así, sin querer, hemos visto dos tipos de buses: el de datos, por donde viajan las informaciones, y el de direcciones, que determina la ubicación. (FIG. Nº 3)

El bus de datos tiene la característica de ser bidireccional; es decir que permite tanto el ingreso de datos como la salida de ellos desde el punto de



vista del microprocesador, mientras que el bus de direcciones es unidireccional. Sólo una telefonista por vez puede discar, de lo contario se produciría una gran confusión.
El 6510 tiene previsto no ser el único elemento que disque, por ello incorpora una llave interna que desconecta a su Registro Contador de Programa del bus de direcciones.

OTROS PINES

Si observamos atentamente la disposición de las patitas del 6510, notaremos que, además de las necesarias líneas de direcciones y de datos, existen otras que se relacionan con nuestro microprocesador y que pasamos a comentar. (FIG. Nº 4)

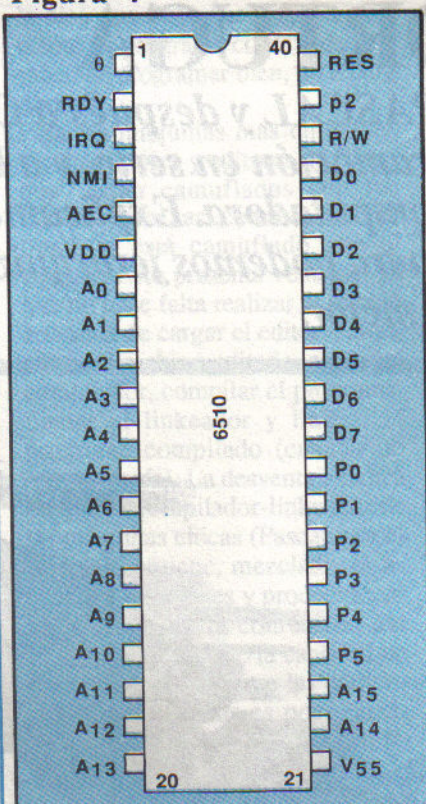
Encontramos una familia de 6 líneas que van desde P0 a P5 y que conforman parte de aquel registro bidireccional que especificamos en la comparación con el 6502.
Cada una de estas líneas tiene una función específica. Las líneas P0, P1 y P2 actúan sobre la estructura de memoria para seleccionar trozos de la misma, mientras que P3, P4 y P5 actúan sobre el control del datasete.
El registro se completa considerando las líneas NMI* y RDY.
La línea RDY controla el funcionamiento del microprocesador. Un cambio de estado en este bit detiene la ejecución del mismo permitiendo, entre otras cosas, un acceso directo a la memoria desde el exterior.
Cuando la línea READY esté en "1" el microprocesador funcionará normalmente, pero el estar en "0" este se detendrá (ante la siguiente operación de lectura), llevando al sistema completo de buses a un tercer estado de alta impedancia. Esta señal proviene de una And entre la línea DMA* y BA (bus available) que proviene del PLA 82S100.
Dejamos para más adelante la línea NMI*.
Otra de las patitas es la RESET*. Cuando se resetea esta línea lleva al microprocesador a la posición de lectura, cargando en el contador de programa la primera dirección del sistema operativo.
La patita IRQ* (Interrup ReQuest) trae un llamado de los periféricos

cuando estos quieren enviar un mensaje. Esta solicitud está condicionada por el estado del bit IRQ perteneciente al registro de estado.
Ahora sí, veamos la línea NMI* (Non Mascarable Interrupt). Es similar al IRQ* excepto en que produce la interrupción sin consultar nada. Se activa al existir fallas o al pulsar la tecla restore, y ataca al microprocesador y a uno de los CIA 6526.
Estas últimas tres líneas, que podríamos agrupar bajo la denominación común de líneas de interrupción, producen al activarse un alto en la operación del 6510, forzando la carga en el Contador de Programa de la dirección del vector de interrupción correspondiente. Este vector no es la interrupción sino la dirección de inicio de esa rutina.

CUADRO DE VECTORES

Nos quedan aún algunas patitas más. 00 es la línea por donde ingresa la señal de reloj. Todas las operaciones

Figura 4



del microprocesador requieren un patrón temporal que actúe sobre sus circuitos internos. Este patrón temporal determina el tiempo que demorará el 6510 en ejecutar una instrucción y permite, además, una sincronización con el resto de las operaciones dentro de la computadora.
La línea AEC se encarga de corroborarles a los distintos elementos de la Commodore que la dirección presente en el bus de direcciones es válida.
Por último la línea R/W* (Indicación de operación de lectura o de escritura) es la resposable de indicar el sentido de la informaciónque viaja por el Bus de DATOS, controlando por un lado la frontera bus interno-externo del microprocesador y por el otro, las operaciones en las memorias.
De esta manera concluimos la primera parte de nuestra visita al pequeño coloso que vive dentro de la 64.

GUILLERMO FORNARESIO

TRUCOS

RESETEO INVERSO

A través de estos POKEs lograremos un reseteo de nuestra computadora por programa.
Lo podremos utilizar como un método de salida cuando deseamos culminar con la ejecución de un programa

```
10 REM RESETEO
DISPERSO
11 REM RESETEA LA
COMPUTADORA
12 REM Y, PARA
USUARIOS DE FAST LOAD
13 REM NO SE ACTIVA DE
NUEVO
14 :
15 :
16 :
20 REM C-128:SYS 65341
30 REM C-64:SYS 64738
```

PROTECCION DE GRABACION

Este pequeño programa de Jorge Franco, para la Drean Commodore 64/C, utiliza los comandos de la disquetera y para usarlo solo basta tipearlo y seguir las instrucciones. Con él se pueden proteger los disquetes contra el borrado involuntario, pero atención, que una vez protegido ya no es posible grabar en el mismo. Esto es de gran utilidad para el caso de que tengamos guardada información muy importante en un disquete.

```
100REM ----------------------------
110REM PROTECCION DE GRABACION
120REM ----------------------------
130A$=CHR$(233)
140PRINTCHR$(147)
150PRINT"PROTECCION (S/N)?"
160GETX$:IFX$<>"S"ANDX$<>"N"THEN160
170IFX$="N"THENA$="N"
180PRINT"INSERTAR DISKETTE Y PULSARRETURN!"
190GETX$:IFX$<>CHR$(13) THEN190
200OPEN 15,8,15
210PRINT#15,"ID"
220OPEN 2,8,2,"#"
230PRINT#15,"M-W"CHR$(1)CHR$(1)CHR$(65)
240PRINT#15,"U1 2 0 18 0"
250PRINT#15,"B-P 2 2"
260PRINT#2,A$;
270PRINT#15,"U2 2 0 18 0"
280CLOSE2
290PRINT#15,"ID"
```

BASIC TORTUGA

El paso de BASIC a PASCAL y después a C es uno de los factores que inducen a la programación en serio, y a la toma de conciencia del potencial de una computadora. Explicamos las características de cada uno y de qué manera podemos jerarquizar nuestro lenguaje en función de programar mejor.

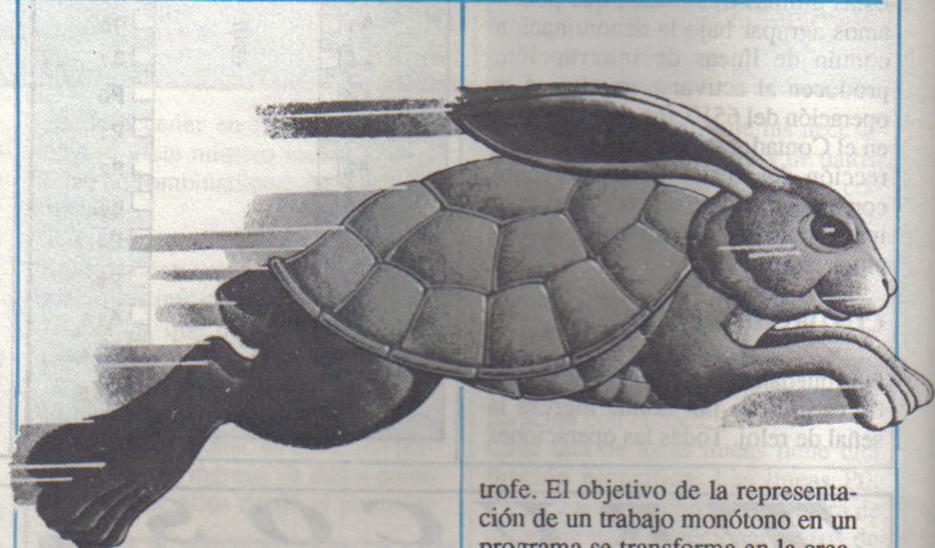
Cuando los usuarios más audaces se aburren de BASIC, la energía oculta de los procesadores consigue atrapar sus mentes precoces.

En ese momento comienza la seducción de cristal: una enfermedad que ataca a los domadores de computadoras que ya han domesticado al lenguaje básico y parásito. Según dicen las viejas, del cristal de los procesadores emanan constantemente ondas atractivas que, según las características del usuario, desembocan en un proceso de adicción al Assembler en diferentes magnitudes.

Esta historia es my común en todos los usuarios piolas, desinformados y con ansias de progreso. La lentitud de BASIC los lleva a alistarse en la búsqueda de algo más atractivo y gratificante. Algo que les permita programar todo lo que desean (casi todo...) que, por supuesto, tiene que correr a velocidades vertiginosas. Por desgracia, con Assembler, el rumbo de sus vidas cambia para peor. No voy a negar las virtudes del código de máquina pero considero que si un programador no profesional tiene una escala de valores, en los niveles de desaprobación máxima de la misma debe estar la programación en esa cosa. El problema que se plantea radica en un cambio de objetivos en la programación que acontece al enfrentarse con tan seductor lenguaje.

EL INICIO DE TODO

Cuando un usuario tiene ya poca suerte y necesita programar su computadora para una tarea específica, se preocupa por la manera en que va a traducir el idioma de sus acciones y pensamientos en un



lenguaje más eléctrico. Y cuando el humano decide acercarse al código de máquina lo hace, como ya se dijo, en busca de velocidad. Esto no constituye una obsesión ya que, cuando se corre más de cinco veces, la mayoría de los programas verifican la relativa dilatación del tiempo. Sobre todo si sus amos los usan para trabajar o estudiar. Frente al problema real, nuestro usuario piola descubre que, si bien en el mundo del procesador todo corre a la velocidad de la luz, las herramientas con las que construía su programa anteriormente se han atomizado en instrucciones que no representan la esencia de las acciones involucradas en la resolución de un problema. Se extrañan mucho cosas como "print sin (x)" después de una semana en las playas del Assembler, cuyos ladrillos de construcción tienen tamaño microscópico en comparación con los de BASIC.

Y finalmente sobreviene la catástrofe. El objetivo de la representación de un trabajo monótono en un programa se transforma en la creación de herramientas para que, después de haberlas terminado, el programa recién comience a tomar forma. La degeneración del objetivo se hace evidente cuando nos percatamos de que hacer un "print sin (x)" desde Assembler implica un tiempo de programación cien veces superior al que supuestamente "perderíamos" si corriésemos nuestro programa en el magnífico BASIC tortuga. Esto quiere decir que vamos a perder más tiempo programando en Assembler para que un programa corra más rápido, que usando su versión lenta.

OTROS LENGUAJES

PASCAL (presentado en notas anteriores) nos ofrecía una posible salida.

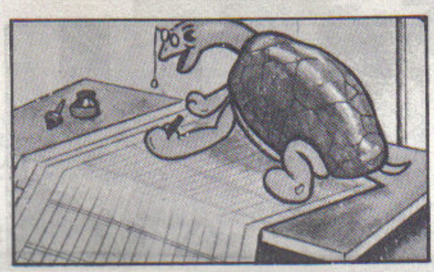
Una manera mucho más elegante, fácil y rápida de programar. Depuración sencilla. Y al momento de la verdad, mayor velocidad. PASCAL es el lenguaje ideal para la gente que quiere programar en serio (estudiantes, por ejemplo) y desa-

rrollar programas para actividades específicas. Pero como siempre hay inconformistas, para algunos PASCAL también puede llegar a resultarles lento. Y antes que caer en el extremo del Assembler, preferiría presentarles algo más moderno.

C también es un lenguaje compilable. Un compilador es un programa que transforma las instrucciones de un "listado" de un programa en rutinas de Assembler. Así, primero se escribe un programa, y después se lo compila para crear un archivo en código de máquina que corre independientemente del compilador y hace lo que nosotros pusimos en ese listado. Salvando las distancias, un compilador BASIC hace justamente eso: transforma las instrucciones de un lenguaje en rutinas en Assembler. Nosotros tenemos que preocuparnos por escribir programas que los humanos entiendan. Después, el compilador interpreta el listado y "pega" rutinas en Assembler que corresponden a las instrucciones del lenguaje aparecidas en el listado.

Las computadoras más grandes, en casi todos los lenguajes compilables, suelen crear en la compilación de programas algunos archivos intermedios que luego no tienen uso. Inclusive la compilación no suele ser el fin del calvario entre escribir un programa y verlo correr. Muchas veces se necesita "linkear" el archivo compilado con un programa llamado "linker". Cuando hace falta este paso adicional, el compilador sólo transforma parcialmente el listado (llamado archivo fuente) en un conjunto de directivas para el linkeador, además de controlar su sintaxis. Las directivas suelen ser, por lo general, los nombres de las rutinas que el linkeador deberá juntar para llegar al archivo ejecutable. Estas rutinas se hallan en otro archivo llamado "librería", que contiene todas las funciones que el compilador implementa. Resumiendo: el compilador lee el listado del programa y lo transforma en un conjunto de órdenes que indican qué rutinas y de qué manera deberán ser pegadas para obtener el programa final. El linkeador lee el archivo compilado, y corre a buscar las rutinas que este le dice que tiene que pegar. Finalmente, el programa cobra vida y, si sabemos programar bien, no tendrá errores.

Para las máquinas más chiquitas, los linkeadores y editores de "listados" están camuflados con los compiladores, así como el sistema oprativo está camuflado con el BASIC. Esto presenta ventajas, ya que no hace falta realizar el trabajo artesanal de cargar el editor, editar, salvar el archivo editado, cargar el compilador, compilar el programa, llamar al linkeador y linkear el programa compilado (cuando no hay otro más). La desventaja radica en que el compilador-linkeador de las máquinas chicas (Pascal, por ejemplo) contiene, mezclada, la librería de funciones y procedimientos a pegar en la confección del programa. Dado que la cantidad de memoria de que gozan las computadoras domésticas es poca, la librería solo permite un número li-



mitado de rutinas (y también un compilador y linkeador poco eficiente). Si bien es más fácil trabajar con ellos, se sacrifica eficiencia por comodidad de programación, y precio de máquina y compilador. En máquinas más grandes, las bibliotecas suelen ser más gordas que los compiladores y linkeadores mismos. Incluso pueden ocupar más de un disco y sobrepasar la capacidad de memoria de la máquina. En estos casos es imposible mezclar la librería con el linkeador y el compilador.

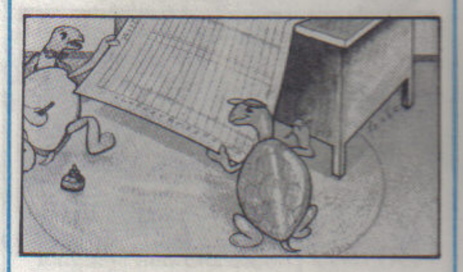
Dadas estas herramientas de programación, resulta evidente que realizar una pequeña modificación de un pequeño programa desemboca en una inmensa tarea de carga, descarga y espera. Por lo general, cuando se trabaja en este tipo de cosas, conviene programar primero en lápiz y papel, tener las ideas claras, y hacer los programas serios cuyos frutos merezcan la pena. La programación-pasatiempo de la mayoría de los usuarios de home computers no se adapta a esta metodología de trabajo. Un caso típico es el de aquel que, no teniendo nada que hacer, se sienta frente a la computadora "para ver si se le ocurre algo" y programa sobre la máquina con el maldito método de ensayo-error. Este consiste en tipear unas líneas que, para ser verificadas como correctas, deben ser corridas, porque tienen tan poco orden que no las entiende nadie. Por lo general esta falta de orden se debe a la proliferación de GOTOs, a la misma escencia del BASIC, y a la increíble capacidad de ocio del dueño.

LEYENDO UN PROGRAMA

Veamos cómo se aplican todos estos conceptos a un compilador C. Por un lado, a diferencia de Pascal, sin hacer distinción entre máquinas chiquitas o grandes, casi todo sistema C cuenta con un editor de textos en el que se escribe el programa, un compilador, un linkeador, y una biblioteca o librería de funciones. Por otro lado, la magnitud de la biblioteca depende de la calidad del compilador y de la máquina. Esto quiere decir que la metodología de trabajo será muy similar en todas las máquinas, pero que los resultados serán dispares conforme al sistema en cuestión (recordemos que el resultado dependía de las funciones de la librería y de la calidad del compilador que, por lo general, se consigue a expensas del tamaño y precio).

Pero para los usos comunes, con funciones matemáticas y gráficas de las más sencillas basta y sobra. De modo que el usuario de una computadora chica no notará la diferencia de rendimiento entre su compilador C y el de una IBM o Apple Macintosh, porque sus exigencias son menores que las de la gente que trabaja con esas máquinas y así les interesa que las cosas sean casi perfectas. En otras palabras, esa diferencia de posibles funciones no representa una atadura de manos del programador aficionado en un compilador chico dado que sus necesidades no son tan terribles. Esto también se ve favorecido por la riqueza del lenguaje. El

C estándar incorpora sentencias simples y de una flexibilidad asombrosa que permiten desde el vamos trabajar de una manera plástica. Por malo que sea un compilador C, mientras sea estándar, posibilitará una infinidad de matices y posibilidades de programación. No obstante el calvario de la programación (edición, compilación, linkeado, ERROR, todo de vuelta) existente en todo compilador C, semejante mazoquismo realmente vale la pena.
C apunta hacia la programación estructurada. La diferencia fundamental entre la programación lineal llena de GOTOs típica del BASIC y el Assembler, y la estructurada se ve fundamentalmente en los listados; un listado de código de máquina se tiene que seguir en el papel para encontrar el camino de salida y un listado de C (o Pascal) cuenta por sí mismo lo que hace; literalmente uno "lee" el programa, o mejor dicho, lee la manera a través de la cual pensó que su monótono trabajo iba a hacerse más mecánico. Se pueden seguir procedimientos y funciones con los nombres más caprichosos a los cuales se les pueden pasar parámetros. Es decir, se pueden inventar instrucciones a medida. Las funciones y procedimientos pueden ser compilados por separado y de esa manera crear una librería propia de funciones. Las variables, al igual que en Pascal, deben ser declaradas con nombre y tipo (reales, enteros, caracteres, arreglos...); pueden ser locales a cada función o globales.
En la escala de altura, C es un lenguaje de bajo nivel. Esto implica muchas cosas. No contiene operaciones para trabajar directamente con entidades compuestas como ser cadenas, arreglos, etcétera. Tampoco cuenta con instrucciones de entrada-salida como print o input, o sentencias de manejo de archivos. Ni siquiera funciones matemáticas o sentencias gráficas. Todas esas pretensiones de "alto nivel" se satisfacen con las instrucciones de la librería. Un lenguaje será de menor nivel en la medida que se parezca al código de máquina en el manejo de memoria y tenga instrucciones no especializadas que hagan "poca cosa", es decir ladrillos pequeños para programar. Los objetos básicos (lo que en BASIC serían variables numéricas, alfanuméricas, archivos) en C son los caracteres, número de punto flotante y enteros de varios tamaños. Por suerte están presentes las cuatro operaciones fundamentales. Las instrucciones son pocas pero tan flexibles y sencillas que la dificultad no está en aprender qué hace cada una, sino en cómo se puede moldear para obtener las extravagancias más grandes. Dado que las instrucciones son muy simples se traducen a pocas líneas de cualquier Assembler (gracias al bajo nivel del lenguaje). En C es posible transportar los códigos fuentes (listados) de una máquina a otra muy fácilmente. Esto implica trabajar lo suficientemente cerca del Assembler como para que el universo corra rápido, y lo suficientemente lejos como para que el programa sea transportable y no dependa de la máquina o del procesa-



dor. Esta característica se denomina transportabilidad.
Para dar una idea de lo compacto del lenguaje veamos algunos números: de las 28 palabras reservadas (el compilador entiende sólo 28 palabras, aparte de los signos) casi la mitad corresponden a las usadas en las declaraciones de los tipos de datos, y el resto a las sentencias de control de flujo. Las funciones de alto nivel son fundamentalmente otros programas en C tomados de la librería que el compilador lee en su momento y pega en el archivo objeto. Son funciones que el fabricante tipeó o ideó por nosotros.
La sintaxis e instrucciones disponibles se encuentran presentes en todos los compiladores de C estándar y con los mismos comportamientos. La diferencia de rendimiento entre los diferentes sistemas se origina en las distintas librerías y en la calidad del compilador y el linkeador. Pero la estructura del lenguaje está presente en todos, y eso ya es bastante.
Cuenta con las sentencias de control de flujo fundamentales para escribir programas estructurados, reunión de sentencias en bloques, toma de decisiones, y bucles muy flexibles. Cuenta con el maldito GOTO, pero al igual que en Pascal, siempre es posible prescindir de él (la proliferación de esta sentencia convierte a los programas en galletas de flujo ingobernables). Dichos saltos se pueden hacer mediante etiquetas ya que no posee (por suerte) números de línea.
Tiene dos defectos fatales: la sintaxis (uso correcto de los signos) y la concordancia de tipos de datos en las asignaciones y en los parámetros de funciones (a una función de números reales no se le puede pasar como parámetros números enteros). Al menos, esto es fatal para los acostumbrados a BASIC; pero con el tiempo se supera. Un buen programa en C tiene casi más caracteres correspondientes a la sintaxis que a las instrucciones; es decir, más paréntesis y llaves que palabras. No obstante ello es ventajoso porque implica una definición muy detallada de una acción a realizar que permite acotar en forma aguda el significado de una sentencia y conseguir un código más eficiente.
Un objeto muy importante en C es el puntero. Un puntero es una variable que apunta a una dirección de memoria que, por lo general, contiene otra variable. El puntero puede señalar, por ejemplo, a un número entero sin signo. Un puntero puede apuntar también al primer elemento de un vector. C reconoce el tipo de dato al cual él se está dirigiendo con ese puntero; gracias a esta característica es posible incrementar el puntero de a uno e indicar al casillero siguiente del vector sin preocuparnos por el tamaño de sus elementos. Indexar el subíndice de un vector o sumarle un número entero a un puntero que señala a su primer elemento es lo mismo. En particular una cadena es un vector de caracteres. Si el puntero p apunta al comienzo de la misma, p+5 apuntará al elemento 5 de la cadena. Por medio de una operación es posible saber cuál es

el contenido del lugar al que apunta un puntero. Un puntero puede señalar a otro puntero. E inclusive a un procedimiento definido por el usuario. Si defino una función a la que le debo pasar un vector de letras para que las ordene alfabéticamente, puedo hacer que la función reciba también un puntero que apunta a un procedimiento que las ordene. Como yo puedo cambiar el valor de un puntero, puedo hacer que este indique un procedimiento de ordenación en particular, e invocar a la función con el vector de letras y ese puntero. Y si más tarde aparece otro vector de letras a las cuales podría ordenar más eficazmente otro procedimiento, invoco a la misma función, pero con el puntero señalando al otro procedimiento.
En la próxima nota veremos ejemplos concretos de programación en C.

Alejandro Parise

RESOLUCION DE SISTEMAS DE ECUACIONES LINEALES

Este programa utiliza el método de eliminación de Gauss para resolver sistemas de ecuaciones lineales de cualquier número de incógnitas.

Comp.: Drean Commodore 64/C - Commodore 128
Conf.: Básica
Tipo : Utilitario
Autor: Carlos G. López

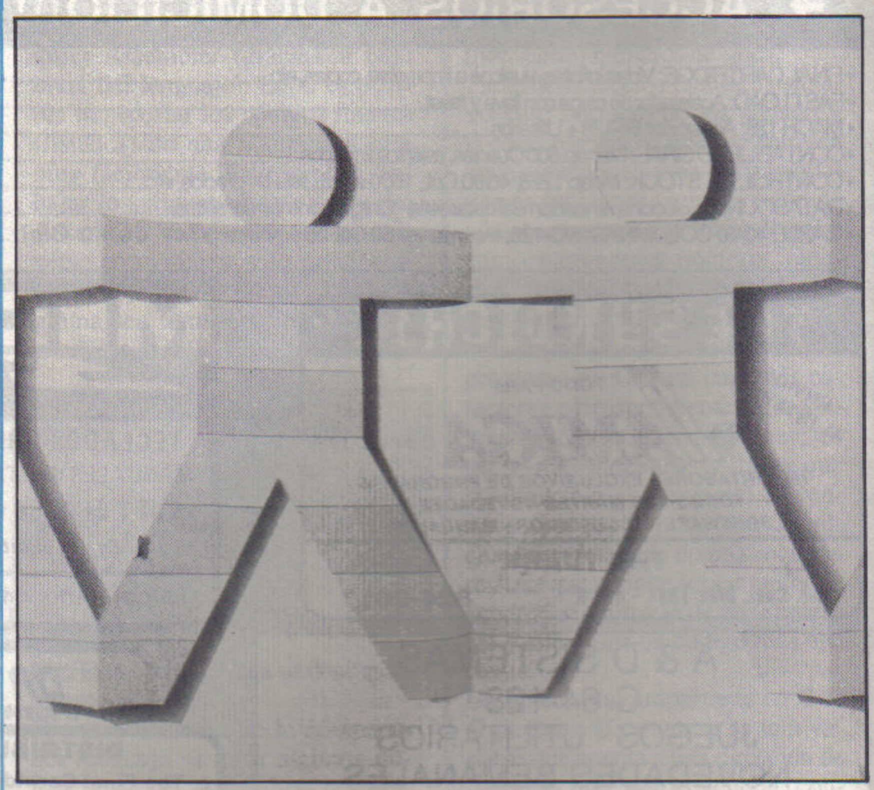
Los sistemas de ecuaciones lineales aparecen de manera natural en casi todas las ramas de la matemática, tanto pura como aplicada. En esta última son innumerables los problemas de física, ingeniería, economía, que conducen a tener que resolver uno o más sistemas.

Pero, ¿qué es un sistema de ecuaciones lineales? Para mayor simplicidad, estudiaremos un ejemplo que involucre solo dos incógnitas. La esencia del proceso es la misma para un problema con un número de incógnitas mayor. Veremos el clásico problema de encuentro entre dos móviles. Hay un móvil, que llamaremos 1, que parte de la ciudad A con una velocidad V1 conocida, en dirección a la ciudad B, distante una distancia L, también conocida, de A. Desde B parte simultáneamente un móvil que denominaremos 2, en dirección a A y con velocidad V2 conocida. Nuestras incógnitas, lo que deseamos conocer, son las distancias recorridas por cada móvil en el momento en que se cruzan en el camino. A la distancia recorrida por 1 la llamamos X1 y a la recorrida por 2, X2 (VER FIGURA 1 y FIGURA 2).

Para poder calcular X1 y X2 debemos plantear ecuaciones que las vinculen; son las ecuaciones del sistema. Para este caso:

X1 + X2 = L
V2*X1 - V1*X2 = 0.

Este es un ejemplo de un sistema de dos ecuaciones con dos incógnitas.



Se resuelve en forma manual y fácilmente con el método de los determinantes. Todos alguna vez lo hemos hecho en el colegio secundario. Pero cuando el sistema es de tres ecuaciones con tres incógnitas, la resolución se hace bastante más engorrosa. Para un número de ecuaciones mayor a tres, salvo casos muy especiales, la resolución es tan fatigosa que no vale la pena intentar una solución manual, porque es altamente probable que nos equivoquemos en algún paso. Y en algunas ramas de la matemática, por ejemplo en la teoría de las ecuaciones diferenciales, es normal la aparición de sistemas con un número enorme de ecuaciones.

EL METODO DE ELIMINACION

Para presentar la mecánica del método trabajaremos con un sistema de tres ecuaciones con tres incógnitas. Por supuesto todo lo dicho para este caso particular tiene validez en un caso más general. Recordemos que llamamos miembro a todo lo que está a cada lado de la igualdad y término a todo lo que está separado por un símbolo de suma y resta (VER FIGURA 3).

La esencia del método está basada en dos propiedades de las igualdades:

1. Si se multiplican ambos miembros de una igualdad por un mismo número, sigue habiendo igualdad.
2. Si a una igualdad le sumamos o

restamos miembro a miembro otra igualdad, también continúa la igualdad. A partir de estas dos propiedades veremos coómo podemos hace para solucionar el sistema.

$$A(1,1)X(1) + A(1,2)X(2) + A(1,3)X(3) = B(1)$$
$$A(2,1)X(1) + A(2,2)X(2) + A(2,3)X(3) = B(2)$$
$$A(3,1)X(1) + A(3,2)X(2) + A(3,3)X(3) = B(3)$$

La letra A designa números conocidos, lo mismo que la B. En cambio la X designa las incógnitas. El primer subíndice de la A indica la ecuación en la que está, y el segundo la incógnita a la que multiplica. Por ejemplo A(2,3) es el número que multiplica a la incógnita 3, X(3), en la segunda ecuación. Los B son los llamados términos independientes, es decir que no multiplican a ninguna incógnita. Definimos ahora dos coeficientes M2 = A (2,1) / A (1,1) y M3 = A (3,1) / A (1,1) Multiplicamos la primera ecuación por M2 y le restamos la segunda ecuación. Luego multiplicamos la primera ecuación por M3 y le restamos la tercera ecuación. El lector que se tome el trabajo de efectuar estas operaciones verá que la incógnita X(1) desaparecerá de la segunda y tercera ecuación. Los coeficientes A y B cambian, pero por las propiedades 1 y 2 sabemos que las

Figura 1 INSTANTE INICIAL

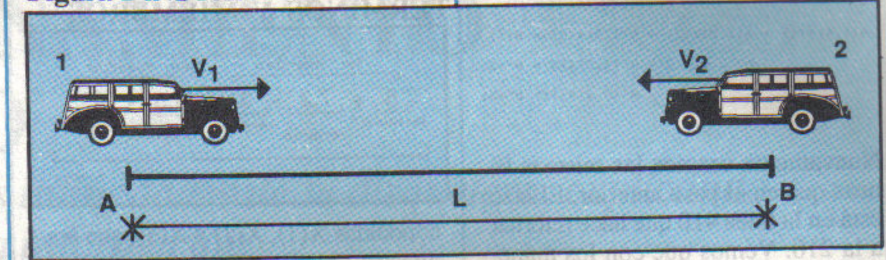
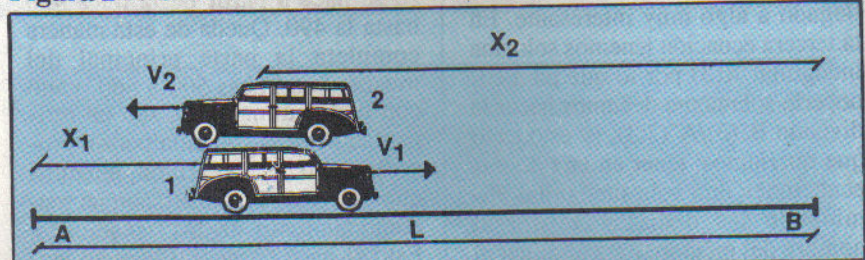


Figura 2 INSTANTE DEL ENCUENTRO



igualdades se mantienen. En el programa esto está hecho a partir de la línea 320. Trabajamos definiendo uno solo de los coeficientes M por vez. De esta manera utilizamos las mismas líneas de programa para todos los M. Esto lo conseguimos mediante un lazo. Luego de redefinir todos los coeficientes A de una ecuación (líneas 330-370), definimos el coeficiente M correspondiente a la ecuación siguiente, en la que repetimos el proceso.

El sistema de ecuaciones queda ahora:

$$A(1,1)X(1) + A(1,2)X(2) + A(1,3)X(3) = B(1)$$
$$A'(2,2)X(2) + A'(2,3)X(3) = B'(2)$$
$$A'(3,2)X(2) + A'(3,3)X(3) = B'(3)$$

Repetimos entonces el procedimiento definiendo M3' = A' (3,2) / A' (2,2). Multiplicamos la segunda por este coeficiente y le restamos la tercera. Desaparece así X(2) de la tercera ecuación. Ahora el sistema es:

```
10 REM ****************************
20 REM *                          *
30 REM *  RESOLUCION  DE SISTEMAS *
40 REM *                          *
50 REM *       DE ECUACIONES      *
60 REM *                          *
70 REM *  POR CARLOS LOPEZ        *
75 REM *                          *
80 REM *       Y FERNANDO PEDRO   *
90 REM *                          *
95 REM ****************************
100 PRINT CHR$(147)
110 INPUT "NUMERO DE INCOGNITAS";N
120 INPUT "ERROR DESEADO";MX
130 DIM A(N,N),B(N),X(N),H(N,N),T(N),R(N),Z(N),Y(N)
140 FOR I=1 TO N
150 FOR J=1 TO N
160 PRINT"A("I","J")":INPUT A(I,J):H(I,J)=A(I,J):NEXT J
170 PRINT"B("I")":INPUT B(I):Y(I)=B(I)
180 NEXT I
190 FOR I=1 TO N:Z(I)=0:NEXT I
200 K=1
210 I=K+1:L=K
220 IF ABS(A(I,K))>ABS(A(L,K)) THEN L=I
230 IF I=N THEN 250
240 I=I+1:GOTO 220
250 IF L=K THEN 310
260 J=K
270 D=A(K,J):A(K,J)=A(L,J):A(L,J)=D
280 IF J=N THEN 300
290 J=J+1:GOTO 270
300 C=B(K):B(K)=B(L):B(L)=C
310 I=K+1
320 M=A(I,K)/A(K,K)
330 A(I,K)=0:J=K+1
340 A(I,J)=A(I,J)-M*A(K,J)
350 IF J=N THEN 370
360 J=J+1:GOTO 340
370 B(I)=B(I)-M*B(K)
380 IF I=N THEN 400
390 I=I+1:GOTO 320
400 IF K=N-1 THEN 420
410 K=K+1:GOTO 210
420 X(N)=B(N)/A(N,N):I=N-1
430 J=I+1:S=0
440 S=S+A(I,J)*X(J)
450 IF J=N THEN 470
460 J=J+1:GOTO 440
470 X(I)=(B(I)-S)/A(I,I)
480 IF I=1 THEN 500
490 I=I-1:GOTO 430
500 FOR I=1 TO N:X(I)=X(I)+Z(I):NEXT I
510 GOSUB 1000
520 IF KJ THEN 570
530 FOR I=1 TO N:FOR J=1 TO N:A(I,J)=H(I,J):NEXT J:NEXT I
540 FOR I=1 TO N:B(I)=R(I):NEXT I
550 FOR I=1 TO N:Z(I)=X(I):NEXT I
560 GOTO 200
570 PRINT"   SALIDA  DE RESULTADOS "
571 PRINT"   ------------------------ "
572 PRINT"1. POR PANTALLA"
573 PRINT"2. POR IMPRESORA"
574 GET Z$:IF Z$<>"1"AND Z$<>"2" THEN 574
575 IF Z$="1" THEN 580
579 OPEN 1,4:CMD1
580 PRINT "LAS SOLUCIONES DEL SISTEMA SON"
590 PRINT "------------------------------"
600 FOR I=1 TO N
610 PRINT "X("I")="X(I):PRINT
620 NEXT
625 IF Z$="1" THEN 640
630 PRINT#1:CLOSE1
640 PRINT "DESEA CAMBIAR LOS TERMINOS INDEPENDIENTES? (S/N)"
645 GET B$
650 IF B$="N" THEN 700
660 IF B$<>"S" THEN 645
670 FOR I=1 TO N:PRINT "B("I")":INPUT B(I):Y(I)=B(I):NEXT I
680 FOR I=1 TO N:FOR J=1 TO N:A(I,J)=H(I,J):NEXT J:NEXT I
690 GOTO 190
700 END
1000 FOR I=1 TO N
1010 T(I)=0
1020 FOR J=1 TO N
1030 T(I)=T(I)+H(I,J)*X(J)
1040 NEXT J
1050 NEXT I
1060 FOR I=1 TO N:R(I)=Y(I)-T(I):NEXT I
1070 KJ=1
1080 FOR I=1 TO N:IF ABS(R(I))>MX THEN KJ=0:NEXT I
1090 RETURN
```

```
A(1,1)X(1) + A(1,2)X(2) + A(1,3)X(3) = B(1)
             A'(2,2)X(2)+ A'(2,3)X(3)= B'(2)
                          A"(3,3)X(3)= B"(3)
```

Nuevamente usamos las mismas líneas que en el paso anterior. El lazo está en la línea 410 que hace retornar a la 210. Vemos que con los manejos algebráicos anteriores hemos llegado a algo muy interesante. En la tercera ecuación tenemos solo una incógnita: X(3) por lo que para conocerla solamente debemos hacer la división B"(3)/A"(3,3), ¡y esto es lo que queríamos, tener en una de las ecuaciones una sola incógnita, eliminando las demás!. De ahí el nombre del método. Para eliminar el resto de las incógnitas recorremos el camino inverso. Una vez averiguada X(3) la introducimos como dato conocido en la segunda ecuación, que tiene entonces como única incógnita X(2). Averiguada X(2) la introducimos junto con X(3) en la primera descubriendo así X(1). Tenemos así

Figura 3 DIFERENCIA ENTRE TERMINOS Y MIEMBROS

1er. miembro / 2do. miembro

5 + 7.2 - 3 = 4.5 - 4

término · término · término · término · término

conocidas X(1), X(2) y X(3), que era nuestro objetivo. En el programa eso se hace a partir de la línea 420 hasta la 490. Queda de esta manera completa la parte principal del mismo. Las líneas 500 a 560, junto con la subrutina 1000, son un refinamiento de la solución por un procedimiento de interacción, mientras que las líneas 210 a 310 son un reacomodamiento de las ecuaciones. Esto se hace porque, si bien en teoría podemos elegir arbitrariamente a cuál ecuación llamamos primera, a cuál segunda, etcétera, en la práctica, y debido a que la máquina trabaja con un número finito de decimales se demuestra que ciertos esquemas de las ecuaciones producen un error mucho mayor que otros. El lector interesado puede consultar una exposición más extensa de este método en **Métodos numéricos y programación Fortran** de D.Mc Cracken (Ed Limusa) en donde se tratan en detalle los problemas de refinamiento de la solución y reacomodamiento de las ecuaciones, además de otros métodos de resolución.

DESCRIPCION DEL PROGRAMA

110-190: entrada de datos y dimensionamiento de matrices.
200-300: reacomodamiento de las ecuaciones.
310-410: eliminación de incógnitas.
420-490: cálculo de las incógnitas.
500-560: refinamiento de la solución.
570-700: salida de la solución.
1000-1090: subrutina de refinamiento.

Carlos G. López

TRUCOS

ACCESO SELECTIVO A DISCO

La disquetera 1541, a diferencia de otros drives, permite tener acceso selectivo a la información que en ella se encuentra.
Por ejemplo, en el caso del directorio, nosotros podemos pedirle que nos muestre todos los archivos del tipo PRG (programas) o todos aquellos del tipo SEQ (archivos secuenciales).

Para ello solo debemos saber qué información mandar a la disquetera para que ella sepa qué mostrar.

En este truco vemos cómo pedir un directorio con sólo aquellos archivos que sean de un mismo tipo, como por ejemplo PRG.

Se muestran dos formas. En la primera, para los usuarios del FAST LOAD o similares. En la segunda para aquellos que sólo tienen la disquetera.

La "X" que aquí se representa indica el tipo de programa o archivo que se desea visualizar.

Por ejemplo, si queremos ver todos los programas que tenemos, deberemos hacer:

1-Usuarios Fast Load: $*=P
2-Usuarios 1541 solamente: LOAD"$*=P"

```
10 REM ACCESO
SELECTIVO A DISCO
20 REM NOS MUESTRA
TODOS AQUELLOS
30 REM ARCHIVOS QUE
SEAN DEL TIPO
40 REM SEQ, PRG, REL,
USR
45 :
50 REM DESDE EL FAST
LOAD: $*=X
60 REM DESDE C-64:
LOAD "$*=X",8
70 REM DONDE X ES S, P,
R, U SEGUN SEA
80 REM EL TIPO DE
ARCHIVO A VISUALIZAR
```

LINEA POR LINEA

Este truco nos fue enviado por **Nicolás y Esteban Hasenauer** y participa del **Concurso de notas trucos y programas** de nuestra revista.
Si se quiere el listado del programa línea por línea, y no a la velocidad con que se pasa habitualmente, todos sabemos que se puede pulsar la tecla control para cumplir ese propósito. Pero existe otro procedimiento y es el siguiente:

1. Carguen el programa que deseen listar en la computadora.

2. Tipeen SYS 65001 y luego RETURN

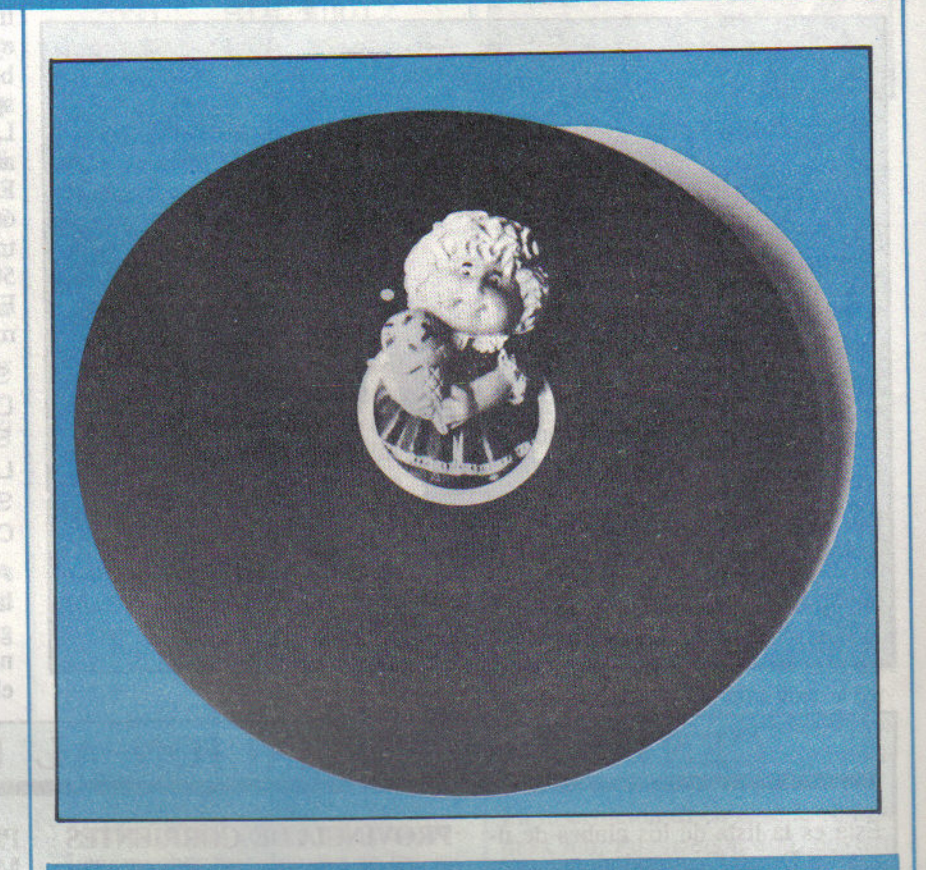
3. Escriban LIST y RETURN

El programa se listará línea por línea. Si quieren desactivarlo sólo pulsen RUN- STOP y RESTORE simultáneamente.
Luego de desactivarlo no se podrá activar nuevamente hasta apagar la máquina.

ALMACENAMIENTO EN DISCO

Les contamos algunas intimidades de la disquetera, explicando cómo se "formatea" la información.

Día a día, los que tienen disquetera realizan una operación casi mística: operar con la disquetera.
Sin embargo no todos saben cómo es el formato o mejor dicho con qué formato se almacena la información en el disquete de nuestro drive.
Por más que tipeemos LOAD o SAVE, nunca sabremos cómo es que todo lo que tenemos en la memoria va a parar a la disquetera.
Sólo sentimos un ruidito, un murmullo, a veces un grito. Todos ellos emanados de alguien que a veces estimamos y que otras, insultamos: el drive.



EL DRIVE

No tiene nada que ver con ese potentísimo limpiador o quitador de manchas de la ropa.
Nos referimos, en este caso, a la 1571 o 1541.
Dentro de él se encuentra alguien aún más importante, el cerebro de la organización.
Su nombre es DOS y su instrumento es el CPU (central de procesamiento) 6502.
Ella es quien determina en donde poner un determinado archivo o dónde almacenar un cierto programa BASIC.
Si tuviéramos la posibilidad de radiografiar a un disco de 5 1/4 pulgadas, es decir el que usa los drive Commodore, veremos en su interior una gran cantidad de cosas extrañas.
Símbolos especiales, caracteres inversos, números romanos, y demás símbolos fantasmales.
Pero, como en todas las cosas, todo tiene su explicación, por más modesta que ésta sea.

DENTRO DEL DISCO

La información se almacena en el disco de acuerdo a los track y sectores que él pueda contener o mejor dicho los que el drive pueda formatear.
Al disquete debemos imaginarlo como realmente es: un disco. Por tal motivo está formado por circunsferencias concéntricas.
A cada una de esas circunferencias se las llama track. Además, dentro de cada track, hay sectores.
Cuando nosotros observamos el directorio del disco, sólo vemos los nombres de los programas y/o archivos que lo constituyen.
Sin embargo, en el disco hay muchos más datos que esos.
Esto es, aquellos que le permiten al DOS (el sistema operativo del disco) saber adónde debe ir a buscar un determinado archivo.
El directorio es un gran índice, que le dice al DOS dónde se encuentra cada uno de los programas que están en el disco.
Esos datos se refieren al tipo de archivo que es (si es programa o si es archivo secuencial o relativo o si es USR).
Además le dice adónde debe buscarlo, es decir a en qué track y en qué sector está.
Por otro lado, en cada track y sector se utilizan dos bytes para decirle al DOS dónde sigue un programa en caso de que no alcance un sector para guardarlo.
En términos técnicos, esto se conoce con el nombre de Link, es decir encadenamiento.

EL HAZ DE BARRIDO

Durante los trabajos de programación suelen surgir determinadas limitaciones, como por ejemplo en el uso de sprites (teniendo como máximo 8) o la activación de la pantalla mitad en alta y el resto en baja, por mencionar algunos.
Este artículo entrega las herramientas necesarias para solucionar esos pequeños inconvenientes cuando se trabaja programando con código máquina.



La Commodore 64 tiene la ventaja de poder controlar el haz de barrido, es decir el conjunto de electrones que conforman la imagen sobre la pantalla del televisor o el monitor.
Al disponer el programador de control sobre este haz, es posible, por ejemplo, disponer tres colores de borde, tres de pantalla, más de 8 sprites y muchas otras opciones.
La movilidad del rayo se efectúa de arriba hacia abajo y de línea en línea. El recorrido se cumple a un ritmo de 60 veces por segundo, cuando se trata del sistema de video NTSC y 50 por segundo si es Pal-N.
El primer paso es efectuar una interrupción IRQ.

```
SEI
LDA  $$--->   Colocación del octeto
              bajo de comienzo
STA  $3014
LDA  $$--->   Colocación del octeto
              alto de comienzo
STA  $3015
CLI
```

A continuación es necesario estudiar la posición de memoria (53273), registro que nos dice si ha ocurrido una interrupción o no. Esto se ve en el bit 0 del registro.

CENTROS DE ATENCION AL USUARIO

Esta es la lista de los clubes de usuarios Drean-Commodore que operan en todo el país.

CAPITAL FEDERAL

Centro de Atención al Usuario Pueyrredón 860, piso 9 (Sede Central); Belgrano: V. de Obligado 2833.

PROVINCIA DE BUENOS AIRES

Lomas de Zamora: Acevedo 48
Ramos Mejía: Bartolomé Mitre 180
La Plata: Calle 50 número 637
Bahía Blanca: Las Heras 81/95
Pergamino: Alem 532
Tandil: Rodríguez 769

PROVINCIA DE CORDOBA

Río Cuarto: Vélez Sarsfield 62

PROVINCIA DE CORRIENTES

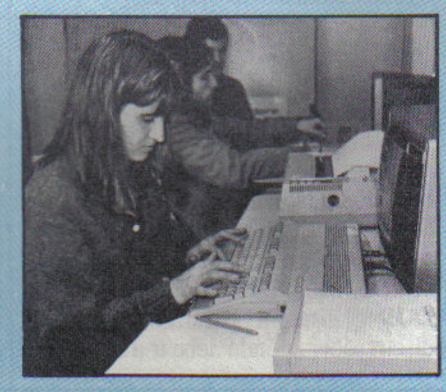
Corrientes: Junín 1327, primer piso "A"

PROVINCIA DE ENTRE RIOS

Concordia: Urquiza 742

PROVINCIA DE LA PAMPA

General Pico: Calle 24 número 433



PROVINCIA DE MENDOZA

Mendoza: San Martín 1052 piso quinto "21"

PROVINCIA DE SALTA

Salta: Avenida Sarmiento 429

PROVINCIA DE SANTA FE

Santa Fe: 4 de Enero 2770
Rosario: San Martín 641

PROVINCIA DE SANTA CRUZ

Río Gallegos: San Martín 1201

PROVINCIA DE TUCUMAN

San Miguel de Tucumán: San Juan 451

CENTRO DE ATENCION AL USUARIO

Por lo tanto debemos realizar un AND §$01 del valor del acumulador tomado de 53273.

```
STA $D019
AND $$01
BNE------> Programa de control
           de barrido
JMP $EA31
```

La función principal de la instrucción STA$D019 es anular la interrupción de barrido si esta ocurre. Se hace poniendo un 1 en el bit de barrido (bit cero); permitiendo cuando hayamos terminado nuestra rutina continuar con otra interrupción.
La siguiente línea comprueba si el valor del Acumulador no es cero, si es así bifurca el resto de nuestro programa. Si no está puesto el bit, salta a la rutina normal IRQ.
Volviendo a la interrupción, ahora debemos comprobar el valor del barrido en la posición 53266, para ver si es nuestra primera o segunda interrupción. Las siguientes líneas lo realizan.

```
LDA $ D012
BMI $ Otro barrido
```

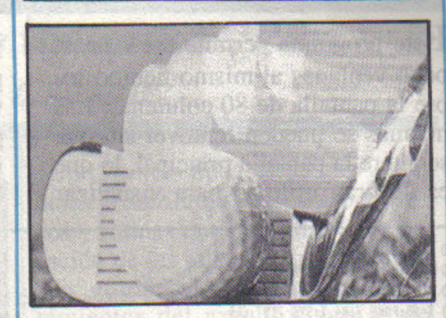
Aquí tenemos la rutina que vamos a utilizar para preparar la interrupción de barrido.

```
2000   SEI
2001   LDA $$1F
2003   STA $3014
2006   LDA $$20
2008   STA $3015
200B   CLI
200C   LDA $$64
200E   STA $D012
2011   AND $$7F
2014   STA $D011
2016   LDA $$01
2019   STA $D01A
201B   RTS
201E   LDA $D019
201F   AND $$01
2022   BNE $2029
2024   JMP $EA31
2026   STA $D019
2029   LDA $D012
202C   BEQ $2041
202F   LDA $$00
2031   STA $D012
2033   LDA $$00
2036   STA $D020
2038   STA $D021
203E   JMP $FEBC
2041   LDA $$64
2043   STA $D012
2046   LDA $$02
2048   STA $D020
204B   STA $D021
204E   JMP $FEBC
```

El JMP $FEBC es la rutina que se usa normalmente al final de una interrupción de barrido. Esto deja las cosas ordenadas y concluye la interrupción.
El programa hace dos interrupciones por cuatro, una para devolvernos al principio y las otras para hacer los cambios requeridos. La posición que se lee es la 53266, que nos dice la línea en la que está en ese momento el rayo de barrido.
Esta es la base de los gráficos por exploración de barrido. Se notarán en el programa ciertas vibraciones, pero no obstante dejo a los investigadores "fanáticos" del código máquina realizar muchos arreglos que no han sido efectuados para no complicar el concepto.

Rodolfo Pansa
Dpto. Técnico Centro de Atención al Usuario de Drean

GOLF PARA COMPUTADORA



El primer campeonato de golf por computadora Drean-Commodore se disputará en la sede central del Centro de Atención al Usuario el sábado 26 de setiembre.
El certamen, que servirá asimismo para confeccionar -de acuerdo con los resultados- el primer ranking de la flamante Asociación de Compugolfistas (ADC) se disputará a 72 hoyos (cuatro vueltas) sobre las canchas del World Class Leader Board Golf.
Durante la prueba los jugadores deberán utilizar el nivel amateur del programa y el sistema de juego será el Medal Play (contra la cancha):
Juez del torneo será Luis Darío Pettina Goobar, del Centro de Atención al Usuario, quien en una charla con la revista Drean-Commodore informó asimismo que cada competidor podrá concurrir con su propio joystick.
La inscripción es en la sede central del Club, Pueyrredón 860, noveno piso, teléfonos 961-6430 ó 962-4689 y está previsto que los partidos comiencen a las dos de la tarde del sábado 26 de setiembre.
¡A preparar los palos de golf y las zapatillas adecuadas!.

CLUB DE USUARIOS

VENTANAS MAGICAS PARA C-128

Este programa permite crear hasta ocho ventanas al mismo tiempo sobre la pantalla de 80 columnas. Las mismas se pueden remover sin que se altere la pantalla principal, lo que es de gran utilidad para visualizar información complementaria. El programa 1 es el generador de las ventanas, y en el 2 damos un ejemplo de cómo usarlo. Este le dará las instrucciones y las diferentes funciones que tiene, como por ejemplo listar el directorio en una de las ventanas. La utilidad del mismo se comprende al ver que el programa puede ser usado tanto en sistemas como en juegos o educativos.

```
10 REM DEMONSTRACION
20 FAST:GRAPHIC 1:GRAPHIC 5
30 BLOAD"CUSTOM WINDOWS",B0,P7169
40 X=16:POKE 7194,X:SYS 7197:CLR
50 COLOR 5,8:WINDOW 2,8,35,17:SYS7200
60 PRINT"BIENVENIDO AL MUNDO DE"
70 PRINT"LAS VENTANAS MAGICAS PARA"
80 PRINT "LA DREAN-COMMODORE C-128"
90 PRINT "ESTE POTENTE PROGRAMA PERMITE"
100 PRINT "CREAR HASTA OCHO VENTANAS AL"
110 PRINT"MISMO TIEMPO SOBRE LA PANTALLA"
120 SLEEP 2:COLOR 5,15:WINDOW 40,12,70,20:SYS 7200
130 PRINT "...SE PUEDEN REMOVER
140 PRINT "Y LA PANTALLA ORIGINAL
150 PRINT "NO SERA MODIFICADA
160 SLEEP 5:COLOR 5,11:WINDOW 15,16,41,22:SYS 7200
170 COLOR 5,2:WINDOW19,14,47,19:SYS 7200
180 COLOR 5,15:WINDOW 24,10,54,16:SYS 7200
190 COLOR 5,4:WINDOW 31,7,60,13:SYS 7200
200 COLOR 5,5:WINDOW 39,4,65,10:SYS 7200
210 COLOR 5,14:WINDOW 45,7,75,14:SYS 7200
220 PRINT"AQUI HAY OCHO VENTANAS":SLEEP6
230 GOSUB 260:GOSUB260:GOSUB260:GOSUB260:GOSUB260:PRINT
240 REM
250 GOTO 280
260 FOR I=1 TO 500:NEXT:SYS 7203
270 RETURN
280 COLOR 5,14:WINDOW 15,10,65,22:SYS 7200:PRINT"TAMBIEN
   PUEDE USAR:"CHR$(142)
290 PRINT TAB(8)"GRAFICOS ASZX"
300 PRINT TAB(8)"COLORES
310 PRINTTAB(8)"REVERSE"
320 PRINTTAB(8)"SUBRAYADO"
330 SLEEP 3:SYS 7203
340 SLEEP1:COLOR 5,8:WINDOW 40,15,65,21:SYS 7200
350 PRINT"SYS 8666 LE PERMITIRA
360 PRINT"LISTAR UN DIRECTORIO
370 PRINT"SOBRE LA PANTALLA.":SLEEP2
380 PRINTTAB(6);
390 SYS 8666:PRINT"DIRECTORIO":CATALOG
400 SLEEP2:SYS 7203
410 COLOR 5,5:WINDOW67,16,77,19:SYS 7200
420 PRINTCHR$(27)"LOPRIMA LA BARRA";
430 X=2500
440 DO:X=X-1
450 GETK$:IF K$=" "THEN X=1:EXIT
460 LOOP WHILE X.0
470 PRINT"":FOR I=0 TO 200:NEXT:SYS 7197:IF X THEN END
```

```
1 REM*****************************
2 REM*                           *
3 REM*   CENTRO DE ATENCION AL   *
4 REM*  USUARIO DREAN-COMMODORE  *
5 REM*                           *
6 REM*   VENTANAS MAGICAS PARA   *
7 REM*   C-128 EN 80 COLUMNAS.   *
8 REM*                           *
9 REM*****************************
10 REM BASIC LOADER
20 GRAPHIC1:GRAPHIC5:FAST:RESTORE
30 FOR I=7169 TO 8689STEP 10
40 FOR J=0 TO 9:READ X:POKE I+J,X:A=A+X:NEXT:NEXT:IFA<>165676THENPRINT"
   ERROR EN DATA":SLOW:END
50 BSAVE "CUSTOM WINDOWS",B0,P7169 TO P8700
60 SYS 7197:END
100 DATA 22,28,0,0,254,37,58,222,49,58,222,53,58,158,55,49,57,55,58,162,0
110 DATA 0,0,0,0,16,0,0,76,48,33,76,54,29,76,15,31,165,228,133,250,165,229
120 DATA 133,251,133,228,173,6,34,141,10,34,234,234,173,0,34,24,105,48,141
130 DATA 71,28,32,125,255,142,19,18,49,146,0,174,1,34,202,32,125,255,183,0
140 DATA 202,208,248,169,112,32,210,255,166,250,134,228,166,251,232,134
150 DATA 229,166,231,134,253,166,230,134,252,134,231,174,2,34,202,208,6
160 DATA 169,19,32,210,255,96,32,116,28,32,125,255,180,0,202,208,248,169
170 DATA 108,32,210,255,234,166,253,134,231,134,230,174,2,34,202,32,116,28
180 DATA 169,170,32,210,255,202,208,250,166,252,232,134,230,166,250,134
190 DATA 229,174,1,34,202,32,116,28,169,175,32,210,255,202,208,250,169,186
200 DATA 32,210,255,32,116,28,32,116,28,164,253,136,166,250,202,24,32,240
210 DATA 255,32,125,255,27,66,0,164,252,200,166,251,232,24,32,240,255,32
220 DATA 125,255,27,84,147,0,174,24,28,134,248,32,180,30,173,27,28,41,128
230 DATA 240,5,169,14,32,210,255,96,32,54,29,169,147,141,74,3,169,1,133
240 DATA 208,96,234,234,169,18,32,45,29,142,1,214,169,19,32,45,29,140,1
250 DATA 214,96,72,169,31,32,45,29,104,141,1,214,96,169,31,32,45,29,173,1
260 DATA 214,96,141,0,214,44,0,214,16,251,96,172,0,34,200,192,9,208,1,96
270 DATA 169,16,24,136,240,5,105,8,76,66,29,133,250,169,34,133,251,56,165
280 DATA 231,229,230,141,1,34,238,1,34,56,233,6,176,1,96,165,230,240,3,238
290 DATA 1,34,56,165,228,229,229,141,2,34,238,2,34,56,233,3,176,1,96,165
300 DATA 228,201,24,240,3,238,2,34,169,0,141,4,34,174,2,34,24,109,1,34,144
310 DATA 3,238,4,34,202,208,244,141,3,34,24,46,3,34,46,4,34,173,9,34,24
320 DATA 109,3,34,141,5,34,173,10,34,109,4,34,141,6,34,176,4,201,255,208,1
330 DATA 96,173,9,34,133,252,173,10,34,133,253,238,0,34,160,0,165,230,145
340 DATA 250,200,165,228,145,250,200,173,1,34,145,250,200,173,2,34,145,250
350 DATA 200,165,252,145,250,200,165,253,145,250,169,0,141,12,34,166,230
360 DATA 240,1,202,138,166,229,240,11,24,105,80,144,3,238,12,34,202,208
370 DATA 245,141,11,34,200,145,250,200,173,12,34,145,250,165,248,141,24,28
380 DATA 169,128,133,248,32,125,255,145,17,0,32,88,30,162,252,142,185,2,32
390 DATA 145,30,173,12,34,24,105,8,141,12,34,32,88,30,32,135,30,32,36,29
400 DATA 141,27,28,32,145,30,173,5,34,141,9,34,76,38,28,24,173,13,34,105,1
410 DATA 144,3,238,14,34,141,13,34,96,173,11,34,141,13,34,173,12,34,141,14
420 DATA 34,96,160,0,162,1,32,119,255,24,165,252,105,1,144,2,230,253,133
430 DATA 252,96,24,173,13,34,105,80,144,3,238,14,34,141,13,34,96,172,13,34
440 DATA 174,14,34,32,8,29,96,173,2,34,141,8,34,173,1,34,141,7,34,32,135
450 DATA 30,32,36,29,32,101,30,206,7,34,208,245,32,120,30,206,8,34,208,228
460 DATA 96,165,230,201,1,240,34,32,88,30,174,2,34,202,142,8,34,32,120,30
470 DATA 32,135,30,32,36,29,41,254,72,32,135,30,104,32,25,29,206,8,34,208
480 DATA 232,165,228,201,23,240,44,32,88,30,174,2,34,202,32,120,30,202,208
490 DATA 250,174,1,34,202,142,7,34,32,135,30,32,36,29,41,254,72,32,135,30
500 DATA 104,32,25,29,32,73,30,206,7,34,208,232,96,172,0,34,208,1,96,169
510 DATA 34,133,251,169,0,133,250,169,10,24,105,8,136,208,251,168,177,250
520 DATA 141,1,34,200,177,250,141,2,34,200,177,250,141,9,34,133,252,200
530 DATA 177,250,141,10,34,133,253,200,177,250,141,11,34,200,177,250,141
540 DATA 12,34,24,32,88,30,32,106,31,173,12,34,24,105,8,141,12,34,32,88,30
550 DATA 32,106,31,32,116,28,76,166,31,234,173,2,34,141,8,34,173,1,34,141
560 DATA 7,34,32,135,30,32,143,31,165,250,32,25,29,206,7,34,208,243,32,120
570 DATA 30,206,8,34,208,226,96,160,0,162,1,169,252,32,116,255,133,250,165
580 DATA 252,24,105,1,144,2,230,253,133,252,96,32,116,28,206,0,34,208,1,96
590 DATA 169,0,133,250,172,0,34,169,8,24,105,8,136,208,250,168,177,250,133
600 DATA 252,200,177,250,133,228,200,177,250,141,1,34,206,1,34,24,165,252
610 DATA 109,1,34,133,253,200,177,250,141,2,34,206,2,34,56,165,228,133,251
620 DATA 237,2,34,133,250,32,116,28,32,116,28,166,251,164,253,202,136,136
630 DATA 24,32,240,255,32,125,255,27,66,0,166,250,164,252,232,232,200,24
640 DATA 32,240,255,32,125,255,27,84,0,32,125,255,19,29,157,0,96,173,26,28
650 DATA 208,1,96,10,10,133,250,169,255,56,229,250,141,10,34,133,54,133,58
660 DATA 169,0,141,9,34,141,0,34,133,53,133,57,96,32,28,32,120,56,32,226
670 DATA 33,144,16,173,21,3,141,96,32,169,91,141,20,3,169,32,141,21,3,88
680 DATA 96,32,97,32,76,101,250,174,25,28,208,3,76,134,32,206,25,28,96,162
690 DATA 32,142,25,28,96,169,63,197,212,208,249,165,211,41,1,240,243,169
700 DATA 134,141,103,32,208,231,169,63,197,212,208,43,165,211,41,4,240,6
710 DATA 32,249,28,76,109,32,165,211,41,2,240,6,32,35,28,76,109,32,165,211
720 DATA 41,9,240,195,41,1,168,32,28,32,152,208,186,169,115,208,202,165
730 DATA 211,41,12,201,12,208,179,166,236,202,134,250,32,116,28,32,116,28
740 DATA 169,50,56,229,250,176,4,169,50,133,250,165,250,24,105,29,168,162
750 DATA 23,24,32,240,255,32,125,255,27,66,0,164,250,162,0,24,32,240,255
760 DATA 32,125,255,27,84,0,32,249,28,208,183,169,0,133,250,169,16,133,251
770 DATA 133,253,173,0,16,160,7,24,113,250,136,208,251,105,10,133,250,109
780 DATA 8,16,133,252,173,8,16,24,109,0,16,170,160,0,177,252,145,250,200
790 DATA 202,208,248,142,8,16,76,63,32,32,250,32,32,125,255,19,19,159,147
800 DATA 0,162,5,160,56,24,32,240,255,32,125,255,27,66,0,162,1,160,22,24
810 DATA 32,240,255,32,125,255,27,84,27,82,153,0,234,32,32,28,32,125,255
820 DATA 14,32,32,195,207,205,205,207,196,207,210,197,32,49,50,56,32,32
830 DATA 195,213,211,212,207,205,32,215,201,206,196,207,215,211,13,32,32
840 DATA 66,89,32,198,82,69,68,69,82,73,67,75,32,199,79,68,68,65,82,68,44
850 DATA 32,40,67,41,49,57,56,54,13,32,32,32,32,32,32,32,57,54,55,54,53
860 DATA 32,32,66,89,84,69,83,32,70,82,69,69,19,19,0,162,1,160,22,24,32
870 DATA 240,255,32,125,255,142,207,19,19,17,17,17,17,17,17,159,0,32,28,32
880 DATA 96,0,32,191,32,169,0,133,208,96,173,20,3,201,91,240,8,173,20,3
890 DATA 141,95,32,56,96,173,21,3,201,32,208,241,24,96,0
```

JUEGOS DE ESTRATEGIA

Ante una jugada nuestra, la computadora responde con otra que puede ponernos en un aprieto o directamente vencernos. ¿Pero cómo hace la máquina para evaluar su táctica a seguir?

Desde el simple Ta-Te-Ti hasta el muy complicado ajedrez, los juegos de salón son verdaderas "batallas" en las que dos o más jugadores compiten buscando demostrar la superioridad de sus estrategias. Por sus características la computadora es capaz de reemplazar a uno de los competidores y, de acuerdo con la capacidad del programador, tendrá un determinado nivel de juego. Es importante destacar que si bien en los juegos que tienen estrategias difíciles de definir, como el ajedrez, el ordenador esta aún por debajo del nivel humano, en aquellos otros en donde hay una estrategia definida y hace falta una gran memoria y velocidad de cálculo podemos decir que "el programa supera a su programador".

Se puede hacer una clasificación de los juegos de estrategia de acuerdo con dos factores. La información que se posee puede ser perfecta (si no hay movidas simultáneas y ambos jugadores conocen en todo momento mientras dura el juego toda movida anterior) o no (cuando no se cumple alguna de las condiciones). La otra distinción posible es la de los juegos que tienen movimientos de azar, es decir, movimientos que no hacen ninguno de los jugadores, sino algún mecanismo de azar externo, por ejemplo el reparto de cartas.

Con estos factores son posibles las combinaciones que ejemplificamos en la tabla 1.

Antes de comenzar un programa de algún juego de estrategia hay que determinar a qué tipo pertenece. Esto nos dará una idea clara de qué factores de evaluación tenemos que ponderar. Analizando el juego en profundidad iremos descubriendo las diferentes estrategias del mismo. Un consejo útil es ir anotando las distintas ideas de estrategias aunque pensemos que sean malas, ya que después se pueden combinar varias estrategias y la que sola era mala, resulta ser muy buena en grupo. ¿Cómo se logra esto? La respuesta a esta pregunta no es una sola, pero mencionaremos aquí a modo de ejemplo la existencia de un polinomio de evaluación, en el que el resultado de cada estrategia es multiplicado por un coeficiente, a criterio del programador, para fomar así el término de esa estrategia en particular. Sumando todos los términos lograremos una evaluación general en que se han tenido en cuenta varias ideas que, sueltas, no harían nada. También se puede hacerlas actuar en forma independiente eligiendo luego una de las opciones. Esto nos permitirá a la hora de programar disponer de más elementos. Vamos ahora a desarrollar estrategias generales para distintos tipos de juegos y, en notas posteriores, iremos dando trucos y rutinas que permitirán la aplicación de las mismas. En los juegos con información perfecta y con movimientos del azar (los juegos de cartas en los que no hay envites, los de dados) suele ser muy útil usar la memoria de la computadora para almacenar los datos disponibles y usar su "fuerza bruta" en los cálculos de probabilidades para las diferentes manos. En estos juegos interviene muy poco la estrategia a adoptar por el segundo jugador. En cambio en los juegos de movimiento simultáneo se debe tener en cuenta esa estrategia. Por ejemplo en el papel - piedra-tijeras, el jugador humano, por lo general, se inclina por uno de los tres objetos en desmedro de los otros dos. El ordenador puede aprovechar esto haciendo que su opción más frecuentemente empleada sea la que derrote a la preferida del rival.

La dificultad de los juegos de información perfecta sin movimientos de azar es descubrir la estrategia correcta. Si lo logramos, podremos asegurar que la computadora jugará bien. En estos juegos de tablero se hace una búsqueda de las distintas piezas jugadas y se efectúa una evaluación relacionándolas entre sí. El último factor a considerar para desarrollar un programa de juegos es el elemento físico. Esto no solo tiene que ver con la presentación del juego en sí. En los juegos de cartas, por ejemplo, debemos observar cómo mezclamos el mazo y cómo extraemos las cartas, dónde se van a almacenar los datos de los diferentes pozos, cartas en la mano de la computadora, cartas en la mesa. De esta manera tenemos expuestos someramente los diferentes tipos de juegos de estrategia que existen, lo que nos brinda la base para ir desarrollando este tema con profundidad.

Fernando Pedró

Cuadro 1

	INFORMACION PERFECTA	INFORMACION IMPERFECTA
SIN MOVIMIENTOS DE AZAR	AJEDREZ	PAPEL PIEDRA TIJERAS
CON MOVIMIENTOS DE AZAR	LUDO	POKER

COMO HACER NUESTROS PROPIOS JUEGOS EN 128

Cuando se compra una computadora con deseos de programar usualmente se está pensando en la realización del juego propio. Aprovechando las posibilidades que brinda el Basic 7.0, les contamos cómo hacer más sencilla la elaboración de un juego.

Seguramente, muchos usuarios de la 128 se habrán imaginado juegos apasionantes y muy originales. Pero en el momento de programar se hacen un lío de subrutinas o variables que después olvidan, y generalmente desisten profundamente desalentados. Aprovechando las posibilidades que nos ofrece el BASIC 7.0, les contamos cómo hacemos nosotros para organizar todo y hacer mucho más fácil, simple y rápida la elaboración de un juego.

Primero hay que dividir las diferentes rutinas que vamos a utilizar. Por ejemplo:

1- Definir los gráficos a usar (Sprites y Pantallas).
2- Inicializar las variables.
3- Activar los Sprites.
4- Recepción de comandos.
5- Actualización de la posición de los Sprites.
6- Verificación de colisiones y/o decisiones múltiples.
7- Visualización de Información del juego -puntaje, vidas, etcétera-.
8- Volver al punto 4.

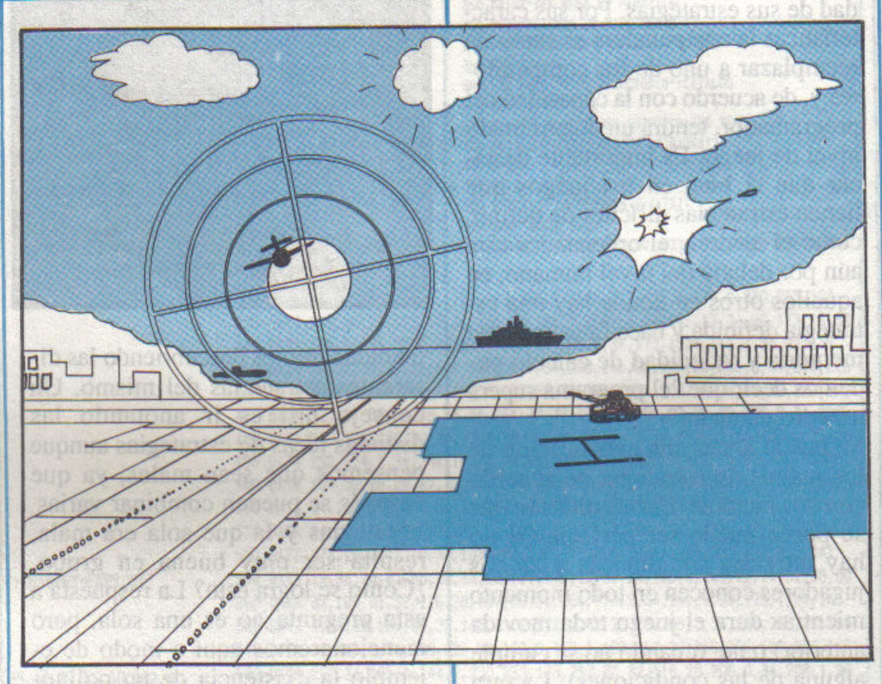
1- DEFINIR LOS SPRITES Y PANTALLAS

Los sprites son fáciles de crear pues el BASIC 7.0 de la Drean-Commodore 128 nos ofrece el comando SPRDEF (SPRite DEFinition) que permite elaborarlos mediante una cuadrícula que amplía 8 veces su tamaño original.

Este editor de sprites posee, además, toda una familia de comandos auxiliares que facilitan ese trabajo.

Una vez terminados los sprites pueden almacenarse en un archivo binario a través del siguiente comando:

BSAVE "nombre del sprite" ,B0,P3584 to P4096



donde el nombre del sprite no deberá exceder los 16 caracteres.

Cuando queramos recuperar el sprite para emplearlo en algún juego, tipeamos:

BLOAD"nombre del sprite"

Pero esto tiene una desventaja, especialmente cuando no queremos usar archivos, por diversos motivos. Para esto tendremos que dibujar en pantalla el sprite, y después transferir esos bytes a las posiciones de los sprites.

SSHAPE A$,x1,y1,x2,y2. **A$**=variable alfanumérica que contiene los bytes copiados de la pantalla.

x1,y1= coordenadas del vértice superior izquierdo.

x2,y2= coordenadas del vértice inferior derecho.

SPRSAV A$,1 graba el contenido de A$ en el Sprite número 1.

Para las pantallas podemos usar DRAW, BOX, CIRCLE, WIDTH, SCALE, PAINT, etcétera.

FIGURA EXPLICATIVA DE SSHAPE Y SPRSAV

```
10 Graphic 1,1
20 Rem dibujo del sprite
30 Draw 1,10,10 to 18,10
40 Draw 1,14,10 to 14,30
50 Draw 1,10,30 to 18,30
60 Rem grabe los pixels
70 SSHAPE A$,10,10,30,31
80 SPRSAV A$,1
90 SPRITE 1,1
100 MOVSPR 1,100,100
```

2- INICIALIZACION DE VARIABLES

En este segundo paso inicializaremos las variables que nos servirán para llevar la cuenta de los puntos, vidas,

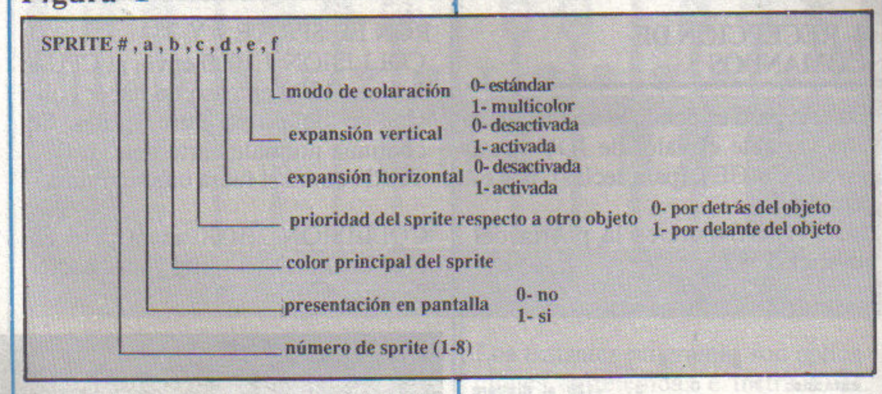
el combustible, el tiempo, etcétera.

3- ACTIVACION DE SPRITES

En esta tercera etapa activaremos los sprites y fijaremos su posición inicial. Podemos usar variables o simplemente podemos usar estas líneas del programa como referencia para líneas posteriores donde se necesitará un reposicionamiento de los mismos.
Las características de un sprite son controladas a través de la sentencia SPRITE mediante los parámetros mostrados en la Figura 1.

Así, por ejemplo, definimos: SPRITE 1,1,1,0,0,0,1:SPRITE 2,1,1,1,1,1,0. Con estas sentencias activamos los sprites 1 y 2 donde el sprite 1 tiene prioridad sobre objetos de la pantalla, no está ensanchado y es multicolor. El sprite 2 no tiene prioridad, está ensanchado y no es multicolor.

Figura 1



MOVSPR 1,100,100:MOVSPR 2,90#3
El sprite 1 está posicionado en las coordenadas para sprites 100,100. El sprite 2 se moverá a 1/5 de la velocidad total, hacia el Este.

Figura 2

```
100 X=100:Y=100:A=50:B=50
110 SPRITE 1,1,1,0,0,0,0:SPRITE 2,1,1,0,0,0,0
120 MOVSPR 1,X,Y:MOVSPR 2,A,B
130 J=JOY(1)
140 IF J=1 THEN Y=Y-1:IF Y<50 THEN Y=250:GOTO 220
150 IF J=3 THEN X=X+1:IF X>320 THEN X=50:GOTO 220
160 IF J=5 THEN Y=Y+1:IF Y>250 THEN Y=50:GOTO 220
170 IF J=7 THEN X=X-1:IF X<50 THEN X=250:GOTO 220
180 IF X<A THEN A=A+1: GOTO 220
190 IF X>A THEN A=A-1: GOTO 220
200 IF Y<B THEN B=B+1: GOTO 220
210 IF Y>B THEN B=B-1: GOTO 220
220 MOVSPR 1,X,Y:MOVSPR 2,A,B
230 GOTO 130
```

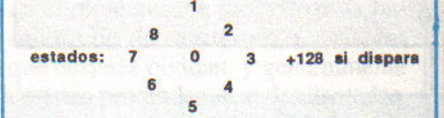
Figura 3

Número que surge al realizarse una colisión	128	64	32	16	8	4	2	1
Número de Sprite	8	7	6	5	4	3	2	1

Al chocar el sprite 8 con el sprite 2 el valor de CH será 130= 128 +2

4- RECEPCION DE COMANDOS

En este paso es donde le asignamos a una variable el valor de JOY (para joystik), o GET (para teclas), por e-jemplo:
J=JOY(port) port es la puerta del joystick 1 ó 2



joystick para arriba estado 1
joystick para la derecha estado 3
joystick para el vértice superior izquierdo estado 8
joystick para abajo + botón de joystick estado 133, etcétera.

5- ACTUALIZACION DE POSICION DE SPRITE

Las nuevas posiciones que tomarán los diferentes sprites podrán ser relativas al punto anterior o no. Esto depende del gusto y del juego de cada uno. Por ejemplo, si se desea que el Sprite 1 se mueva con relación al joystick y que el Sprite 2 lo persiga, podemos emplear un programa como el que se muestra en la Figura 2.

6- VERIFICACION DE COLISIONES Y/O DECISIONES MULTIPLES

En esta etapa verificamos si los Sprites chocan entre ellos o chocan con objetos de la pantalla. Los comandos que utilizamos son COLLISION y/o BUMP.
BUMP (1) es para detectar colisiones entre Sprites.
BUMP (2) es para detectar colisiones entre Sprites y objetos.

Por ejemplo:

```
CH=BUMP(1)
IF CH=3 THEN PRINT "CHOCARON EL SPRITE 1 Y 2"
```

COLLISION 1,1000 envía al CPU a la línea 1000 en caso de haber colisión de algún tipo entre Sprites. Se continúa normalmenrte con un RETURN como si fuera una subrutina.

COLLISION 2,1000 idem pero en caso de chocar un sprite y un carácter de pantalla.



```
100 COLLISION 1,1000
.
.
.
1000 PRINT "CHOCARON DOS SPRITES"
1010 RETURN
```

FIGURA EXPLICATIVA DE COLISIONES

```
10 Sprite 1,1
20 Sprite 2,2
30 A=30 :B=30
40 Movspr 1,150,100
50 J=Joy (1)
60 IF J=1 THEN A=A-1
70 IF J=3 THEN B=B+1
80 IF J=5 THEN A=A+1
90 IF J=7 THEN B=B-1
100 Movspr 2,B,A
110 CH=BUMP (1)
120 IF CH=3 THEN PLAY "A"
130 Goto 50
```

De esta manera cuando choque el sprite 1 y el 2 saldrá un sonido de la máquina.

7- VISUALIZACION DE INFORMACION DEL JUEGO

Esta parte trata de la impresión en pantalla de los puntos, vidas, choques, etcétera.
Esto se facilita con el comando CHAR, pudiendo imprimir en cualquier modo, y en cualquier lugar de la pantalla.

```
100 V=3 REM CANTIDAD DE VIDAS
110 CHAR 1,20,1,"VIDAS";STRS(V)
```

Esto es utilizable para cualquier variable y cualquier posición.

EJEMPLO DE VISUALIZACIÓN

```
1 Graphic 1,1
5 B=0
10 A=INT (RND(1)*10)
20 IF A=5 THEN B=B+1
30 CHAR 1,1,7,"Número"+ Str$(A)
40 CHAR 1,1,10,"veces que la computadora tiró el 5:" +str$(B)
50 Goto 10
```

José Vinograd

MUSICA Y SONIDOS

Comp.: Drean Commodore 64/C
Conf.: Basica
Tipo: Musical
Autor: Alfredo Termeniello

Esta es la continuación de la serie de notas referidas a cómo poder usar la computadora como un instrumento musical cuyo estudio facilitará la introducción del lector en el mundo de los sonidos de la Drean Commodore 64/C. En este caso presentaremos tres programas. El primero de ellos es un ejemplo del A/D/S/R para que el lector se pruebe y se ejercite sacándole sonidos a la máquina. Nació de la necesidad planteada al autor por algunos usuarios que no conseguían hacer variaciones de sonidos.

A/D/S/R son las iniciales de los cuatro términos de la figura 1.

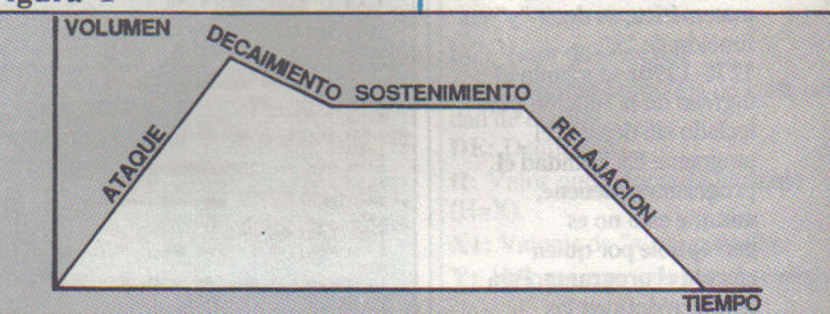
Ataque es el tiempo que tarda el sonido en llegar desde 0 hasta el volumen máximo. Decaimiento es el tiempo desde el volumen máximo hasta que se estabiliza. El tiempo de la estabilización es el sostén hasta alcanzar el relajamiento. Finalmente este representa la última parte de la curva, hasta que desaparece el sonido.

El programa es muy fácil de usar. Hay que modificar donde están las letras A,D,S y R por valores de 0 a 15; luego modificar los mismos para escuchar las variaciones del sonido producidas.

```
1 PRINT" ▣ ▣ " : GOTO 10
2 *********** A/D/S/R *************
3 **** PARA DREAN COMMODORE  ****
4 ***  ALFREDO  TERMENIELLO    ***
5 **        " BARBA'S "         **
6 ***    CALLE 26 NRO. 4218    ***
7 ****          BERISSO        ****
8 ***      LA PLATA (BS.AS.)    ***
9 ********************************
10 POKE 53280,.:POKE 53281,.
20 FOR I=54272 TO 54296:POKEI,0:NEXT
30 POKE 54296,15
45 PRINT" 90 POKE 54277, A*16+D
50 PRINT" 95 POKE 54278, S*16+R
80 PRINT" GT 90 ▣ ":SYS 42115
100 POKE 54276,33
110 POKE 54272,135:POKE54273,33
120 PRINT" ▣ RUN ▣":SYS42115
125 *** 1 CURSOR ↑↑↑ ARRIBA ***
130 INTRODUZCA VALORES DE 0 A 15
140 EN LUGAR DE LAS LETRAS A,D,S,R
150 SI NO CAMBIA ALGUNA,LAS MISMAS
160 ASUMEN VALOR CERO
170 A=ATAQUE          :D=DECAY
180 S=SOSTENIMIENTO   :R=RELAJACION
```

```
20 FOR D=54272 TO 54296:POKE D,0:NEXT
30 POKE 54296,15
40 POKE 54278,0:POKE 54277,9
50 POKE 54276,17
60 FOR N=80 TO 140 STEP 4
70 POKE 54273,N:NEXT
80 POKE 54276,0:POKE 54277,74
90 FOR N=1 TO 150:NEXT
100 POKE 54276,17
110 FOR N=100 TO 120:POKE 54273,N:NEXT
120 FOR N=120 TO 60 STEP-0.4
130 POKE 54273,N:NEXT
```

Figura 1



Los restantes programas son aplicaciones entretenidas e instructivas. Por ejemplo el segundo ejecuta un silbido que nos puede resultar útil si pasa una chica cerca de nuestra computadora.

El último ya es un poco más complicado. Ejecuta una parte de la música de Rambo y tiene un efecto de eco muy bonito producido por la variación de la dirección 54294 con la variable E (línea 112).

```
10 POKE 53280,.:POKE 53281,.:PRINT"▣"
20 Z=54276:GOTO 40
30 X=X+1:RESTORE:GOTO80
40 FOR I=54272 TO 54296:POKEI,0:NEXT
50 POKE 54293,1      :POKE 54294,4
60 POKE 54295,241    :POKE 54296,31
70 POKE 54277,11     :POKE 54278,15
80 READ B,C  :B=B*X  :IF B=0 THEN 30
90 HF=INT(B/256):LF=B-256*HF
91 IF X > 2 THEN X=1
100 POKE54273,HF:POKE54272,LF:POKEZ,33
110 E=100:FOR O=1 TO C*2
112 POKE 54294,E:E=E-10
114 IF E < 2 THEN E=100
119 NEXT:POKE Z,32:POKE 54294,4:GOTO 80
130 DATA 1072,7,1072,3,1072,7,1072,25
140 DATA 803,3,955,3,1072,3,1275,3,1204,3,1072,3,955,3,1072,7,1072,3,1072,7
150 DATA 1072,25,803,3,955,3,1072,3,1275,3,1204,3,1072,3,955,3
160 DATA 851,7,851,3,851,7,851,25
170 DATA 716,3,851,3,1072,3,1275,3,1204,3,1072,3,955,3,851,7,851,3,851,7
175 DATA 851,25,716,3,851,3,1072,3.1275,3,1204,3,1072,3,955,3,.
```

GRAFICADOR DE FUNCIONES

Comp.: Drean
Commodore 128
Clase: Utilitario
Autor: Gerardo Krizan

El programa es un utilitario que sirve para graficar funciones lineales de cualquier tipo. Presenta la ventaja de que al producirse algún error (por ejemplo división por cero, cantidad ilegal, etcétera), no se detiene el programa; además, puede elegirse la escala y la exactitud de graficación (definición).

A continuación se explica el funcionamiento del programa y sus diferentes partes:

Líneas

1000-1090: Carátula.

1100-1210: Instrucciones.

1220-1410: Ingreso de datos.

1260: se ingresa la función, que se almacena en la variable B$. La sintaxis de la función es la utilizada comúnmente en matemática, es decir "y" en función de "x".

1270-1290: se simula el ingreso de la función por teclado sin detener el programa. **En realidad el programa se detiene, aunque esto no es perceptible por quien ejecuta el programa.** Para esto se redefinen las teclas de función F1 y F3. La primera ingresa la línea "1560" con la función, y la segunda continúa la ejecución del programa con un "GOTO 1300".

Si se desea renumerar el programa, cambiar los valores de número de línea de inserción de la función y de reinicio del programa (originalmente 1560 y 1300 respectivamente), que afectan a las teclas de función redefinidas.

1300-1380: se pide el ingreso de la escala y, en base a este dato, se calculan las coordenadas de inicio y fin de los ejes cartesianos dentro de la pantalla, y la longitud en pixels de cada unidad de los mismos. A mayor escala, mayor es esa longitud.

1390-1410: se pide el ingreso de la definición, con lo que se determina la exactitud de graficación. A

```
1000 REM ******************************
1010 REM *                            *
1020 REM *  GRAFICADOR DE FUNCIONES   *
1030 REM *                            *
1040 REM *  (C)1987  GERARDO KRIZAN   *
1050 REM *                            *
1060 REM *  COMMODORE 128  SOFTWARE   *
1070 REM *                            *
1080 REM ******************************
1090 :
1100 REM         *************
1110 REM         INSTRUCCIONES
1120 REM         *************
1130 :
1140 COLOR0,15:COLOR4,7:COLOR1,7
1150 PRINTCHR$(14)"▒▒          ▒ ▁♠▁╮▔♠┳▁ ▔▔ ▁╱╲╮▔╱▔♥"
1160 PRINT"▒KNSTRUCCIONES:"
1170 PRINT"────────────────"
1180 PRINT"              ▔L ─128 LE PEDIRA QUE EN-"
1190 PRINT"▒TRE LA FUNCION. LA MANERA DE  ESCRIBIRLA▒ES LA SIGUIENTE:
1200 PRINT"▒              Y=COS(X)                          O▒ Y=2*X↑2+3"
1210 :
1220 REM       ****************
1230 REM       INGRESO DE DATOS
1240 REM       ****************
1250 :
1260 INPUT"▒▒▒KNGRESE FUNCION ▒";B$
1270 A$="1560 "+B$
1280 KEY1,A$+CHR$(13):KEY3,"GOTO 1300"+CHR$(13)
1290 PRINT"▒▒            ▒▒ULSE -1 Y LUEGO -3▒":END
1300 INPUT"▒▒▒           ▔SCALA  [1-6] ";EC
1310 XA=10:XB=310:YA=10:YB=190
1320 IF EC=1 THEN ES=5:GOTO 1390
1330 IF EC=2 THEN ES=10:GOTO 1390
1340 IF EC=3 THEN ES=15:GOTO 1390
1350 IF EC=4 THEN ES=20:XA=20:XB=300:YA=20:YB=200:GOTO 1390
1360 IF EC=5 THEN ES=25:YA=25:YB=180:GOTO 1390
1370 IF EC=6 THEN ES=30:GOTO 1390
1380 PRINT:GOTO 1300
1390 INPUT"▒            ▔EFINICION [1-10] ";DE
1400 IF DE<1 OR DE>10 THEN PRINT"▒▒▒":GOTO 1390
1410 :
1420 REM        **************
1430 REM        RUTINA GRAFICA
1440 REM        **************
1450 :
1460 FAST:GRAPHIC1,1
1470 FOR T=5 TO 315:DRAW1,T,100:NEXT:FOR T=XA TO XB STEP ES:DRAW1,T,99 TO T,101:
NEXT
1480 FOR T=5 TO 195:DRAW1,160,T:NEXT:FOR T=YA TO YB STEP ES:DRAW1,159,T TO 161,T
:NEXT
1490 SLOW
1500 CHAR1,0,0,B$,1:CHAR1,19,1,"▒":CHAR1,38,13,"▒":CHAR1,26,0,"DEF="+STR$(DE),1:
CHAR1,34,0,"ESC="+STR$(EC),1
1510 WIDTH2
1520 DE=DE*.1:H=160/ES:H=H-.5:FOR X=-H TO H STEP DE
1530 X1=(X*ES)+160
1540 TRAP 1750
1550 REM ** FUNCION **
1560 Y=LOG(1/X↑2)
1570 REM *************
1580 Y1=100-(Y*ES)
1590 IF X1>310 OR X1<10 OR Y1<0 OR Y1>200 THEN X2=0:GOTO 1620
1600 IF X2=0 THEN X2=X1:Y2=Y1:DRAW1,X2,Y2
1610 DRAW1 TO X1,Y1
1620 NEXT X
1630 CHAR1,1,2,"O.K"
1640 CHAR1,1,4,"SPC PARA SALIR",1
1650 :
1660 REM          ***********
1670 REM          S A L I D A
1680 REM          ***********
1690 :
1700 GET A$:IF A$=CHR$(32)THEN GRAPHIC0:ELSE GOTO1700
1710 PRINT"▒▒         ▔ESEA OTRO GRAFICO [S/N] ?"
1720 GETW$:IFW$="S"THEN RUN 1140
1730 IFW$="N"THEN END
1740 GOTO 1720
1750 X2=0:RESUME 1620
```

mayor valor, disminuye la exactitud, aunque aumenta la velocidad de graficación.

1420-1650: Rutina de graficación.

1460-1500: se realiza el trazado de los ejes cartesianos y la división de los mismos en unidades, así como la impresión de los carteles correspondientes a las características del gráfico (función, definición y escala).

1520: se determina el intervalo de graficación de la variable independiente (x).

1530: transforma los valores de "x" en coordenadas dentro del sistema de ejes en pantalla.

1540: se detectan los errores, bifurcándose el programa a la línea 1750.

1560: se encuentra la función a graficar.

1580: se transforman los valores correspondientes a la variable dependiente (y) en valores de coordenadas en la pantalla.

1590: se verifica si el punto a graficar entra o no en la pantalla; en caso que no entre, se continúa el bucle con el siguiente valor de "x".

1600: dibuja el primer punto localizado dentro de la pantalla y aquellos puntos posteriores a valores correspondientes a puntos ubicados fuera de la pantalla. Si no se hiciera así, se produciría la unión de puntos no consecutivos, dando origen a curvas que nada tienen que ver con la realidad.

1610: dibujo de líneas que unen dos puntos consecutivos. Se obtiene así una curva continua y no una sucesión de puntos.

1630-1640: información de fin de graficación.

1660-1740: Salida del programa. Aquí se pregunta si se desea realizar otro gráfico o terminar.

1750: Continúa el programa en la línea 1620 luego de detectarse un error.

VARIABLES UTILIZADAS

B$: Función a graficar.

A$: Número de línea que contiene a la función.

XA: Punto de inicio del trazado del eje "x".

XB: Punto final del trazado del eje "x".

YA: Punto de inicio del trazado del eje "y".

YB: Punto final del trazado del eje "y".

EC: Valor de la escala.

ES: Longitud en pixels de cada unidad de los ejes.

DE: Denifición.

H: Valor que adopta la variable "x" (H=X).

X1: Valores de "x" en pantalla.

Y: Valores que adopta la variable "y".

Y1: Valores de "y" en pantalla.

BOULDER DASH

Comp.: Drean Commodore 64/C
Distribuye: PYM- Soft
★★★★★

En este complejo juego dispondremos de quince cavernas diferentes a elección y, en cada una de ellas, 5 niveles distintos de dificultad.

Las cavernas están rodeadas por inexpugnables paredes de ladrillos, que conforman laberintos en su interior. El Boulder puede moverse hacia arriba, abajo y los costados pero sin atravesar los muros. En su trayecto irá deglutiendo fardos de pasto y abriendo camino hasta llegar a los apetitosos diamantes.

Deberemos tener, sin embargo, mucho cuidado con unos gigantescos cantos rodados, ya que si uno de ellos cae sobre la cabeza de nuestro simpático amiguito, morirá aplastado y, lo que es peor, sin haber logrado saciar su voraz -y exigente- estómago.

Si por el contrario logramos que el Boulder coma todos los diamantes de la primera cueva, sin quedar atrapado por las piedras - o aplastado por una de ellas al caer-, deberemos dirigirlo rápidamente hacia la puerta de salida antes que transcurran en total 150 segundos, y así pasar al segundo nivel.

La cueva número 2 tiene alguna dificultad mayor por el laberinto y, además, por la existencia de un significativo y desagradable monstruito -encerrado entre fardos de pastos- que, muy gráficamente, apenas consiste en dos mandíbulas que rítmicamente se abren y cierran adelantándonos el triste futuro si por error lo liberamos.

La tercera cueva es realmente complicada, ya que, si bien no tenemos ningún monstruo voraz que pueda perseguirnos, lo intrincado del laberinto de ladrillos puede provocar con mucha facilidad que quedemos absolutamente bloqueados y aislados, sin poder mover el Boulder hasta que transcurra todo el tiempo previsto y muera de inanición.

La siguiente pantalla tiene dificultades inesperadas. No existe ningún laberinto, ni diamantes, sino solo cuatro mariposas encerradas, a las que será necesario liberar de a una. Claro que si una de ellas toca al Boulder, este morirá, por lo que deberemos obrar con rapidez, liberar a una de ellas y hacerla seguir nuestro rastro, hasta un punto desde el cual podamos hacer caer sobre ella un canto rodado. El impacto la matará y transformará -¿a qué no adivinan?...- en 9 apetitosos diamantes.

La séptima es una variante que incluye los elementos de las cuatro primeras pantallas, con un agregado siniestro: una lava verde que, poco a poco, va creciendo desde el fondo de la cueva. Al mismo tiempo que tratamos de comer diamantes, de no liberar a algún monstruo, no morir aplastados por un canto rodado y no excedernos en el tiempo, deberemos evitar que la lava se extienda arrojando piedras sobre los resquicios por donde se filtra hacia arriba. Si la lava toca al Boluder, este pasará a mejor vida, y si no lo toca, pero llega a cubrir una porción predeterminada de la cueva, se solidificará en multitud de cantos rodados y, nuevamente, nuestro simpático bichito quedará bloqueado.

Por último les relatamos la novena cueva, en realidad una doble, ya que el Boulder se mueve libremente por debajo -donde no hay nada- pero todo el techo está integrado por toneladas de diamantes y cantos rodados. Con pericia y precisión habrá que comer los dos diamantes que bloquean los únicos agujeros de la caverna, y salir de allí con presteza y rapidez antes de morir aplastados por el alud.

Una vez que este haya concluido... ¡a comer se ha dicho!, comenzando por los que cayeron a la parte inferior -siempre con cuidado- y luego los que aún quedaron arriba, provocando nuevos desmoronamientos para liberar los apetitosos bocados que restan.

Al comenzar a utilizar el Boulder podremos elegir cualquiera de las 15 cuevas y el nivel de dificultad respectivo, así como advertir a la computadora si participarán uno o dos jugadores.

En todos los casos la parte superior de la pantalla nos informará -según la cueva y el nivel de dificultad seleccionado- cuantos diamantes(deberemos deglutir y de cuánto tiempo disponemos.

BONDUELLE SOCCER

Comp.: Drean Commodore 64/C
Editor: The Future Division
Distribuye: The Tuerk
★★★

Llegó la hora de los deportes y, como no podía ser de otra manera, comenzamos por el fútbol. Bonduelle Soccer es un partido entre dos equipos de siete hombres cada uno, enmarcado en la final de la Copa del Mundo de México '86.

El partido se disputa a dos tiempos de 200 segundos de duración cada uno y en un estadio colmado de público que sigue ruidosamente -y con buenos efectos sonoros- cada gol o jugada de emoción.

Ningún detalle de la realidad falta en esta justa de-

★★★★★	EXCELENTE	★★	REGULAR
★★★★	MUY BUENO	★	MALO
★★★	BUENO		

portiva. Los corners, saques de arco y laterales se cumplen a pie ¡puntillas luego de cada estridente llamado del árbitro
Con el joystick dirigiremos nuestro jugador más próximo al balón y, oprimiendo el botón de fuego, lo haremos patear la pelota. El mismo botón de fuego servirá para que nuestro guardavalla efectúe "voladas" que el propio Hugo Gatti envidiría, en procura de evitar un gol adversario.

El único problema del que adolece este juego es la baja calidad de sus gráficos, ya que los jugadorres se parecen más a fantasmas peludos y gorditos -o al abominable hombre de las nieves- que a deportistas hechos y derechos.

Este detalle se compensa sin embargo por lo ágil del juego y por las incidencias.

Sobre el final de cada etapa, automáticamente los jugadores se irán a los vestuarios para descansar -apenas 15 segundos- y volverán a ocupar sus posiciones en el campo de juego, pero cambiando de arco, no sea que alguien se queje porque le tocó con el sol o el viento en contra.

Al término del encuentro una damisela rubia -pero no agraciada, por ese mismo problema de los gráficos- saldrá al encuentro del equipo ganador, formado en el centro del campo de juego, y le entregará a su capitán la Copa del Mundo, ante el delirio de las tribunas. Un detalle, para quienes se remontan en la historia de los jueguitos, este tiene reminiscencias del conocido "Fútbol Internacional".



MILKRACE

Comp.: Drean Commodore 64/C
Editor: The Softguru
Distribuye: The Tuerk
★★★

Seguimos con los deportes y en este caso mediante una carrera de bicicletas que, dividida en 14 etapas, recorre prácticamente toda Inglaterra y buena parte del País de Gales.

Claro que se trata de una carrera de leche. ¿Qué cómo es eso? Pues muy simple. La prueba es larga y nuestro deportista pierde energías continuamente, así que en plena marcha y, mientras evitamos chocar con otros ci-

clistas o caernos en algún bache -por lo visto no son privativos de las calles argentinas-, deberemos ir recogiendo botellas de leche colocadas en los bordes de la ruta.
Cada una de ellas nos permitirá recargar un poco de nuestra energía y, si tomamos la mayoría de las botellas, terminar la etapa.



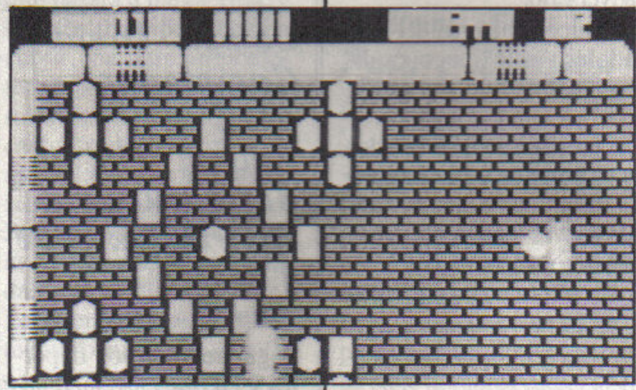
El juego contempla además muchas variantes. Sobre el costado derecho superior de la pantalla veremos un esquema de la pendiente que atravesamos en ese momento. Algunas son realmente muy empinadas y gastaremos muchísima energía para avanzar con lentitud, en especial si estamos tomando velocidad luego de una caída.

Cada vez que "pisemos" la banquina, tropecemos con otro corredor o nos atropelle el furgón que habitualmente "cierra" una carrera de bicicletas, caeremos sobre el pavimento y perderemos preciosos segundos.
Sin embargo aquí no culminarán nuestras penurias, ya que al estar detenidos, deberemos realizar verdaderos malabares para que, mientras todavía circulamos a marcha lenta tratando de ganar velocidad, no nos atropelle alguno de los restantes corredores que avanzan a gran velocidad y que aparecen como un relámpago detrás de nosotros, casi sin darnos tiempo a nada.
Nuestro puesto de largada será el 41, sobre 80 corredores y en los primeros ensayos, invariablemente, quedaremos número 80 en menos de un minuto.

KRAKOUT V2

Comp.: Drean Commodore 64/C
Editor: D.S. Compware
Distribuye: PYM-Soft
★★★★

Un juego entretenido e ideal para cuando nos aprestamos a pasar una larga tarde -o mañana, o noche- solos, debido a su extensión y variedad. Se trata de una mezcla de "flipper", billar y frontón, en el cual una pelotita va golpeando de borde a borde y contra los obstáculos diseminados sobre el tablero.
Sobre uno de los extremos está ubicado nuestro bate que, al ser movido en sentido vertical, permite interceptar la pelotita y devolverla al sector de juego, donde seguirá golpeando -y matando cuadraditos- que representarán más puntos a nuestro favor.
Cada vez que terminemos una pantalla, pasaremos a una distinta. Al efectuar la revisión de este programa, llegamos a la número 80 y sin que ninguna de ellas haya repetido a las anteriores, hasta donde nuestra memoria, a esa altura ya del partido y con tres horas largas para un solo juego, nos permita recordar.
En cada pantalla el diagrama de obstáculos y las exigencias serán distintos. Así algunos cuadrados desaparecen al primer impacto, mientras en otros casos son necesarios 2 golpes, a veces tres, quizás 7 ó 10, y en algunos casos no desaparecen nunca.
Otros, al tocarlos con nuestra pelotita se transformarán y dibujarán una letra sobre su superficie. Cuando los volvamos a golpear, el resultado puede ser diferente.
En alguna oprtunidad "cerrará" la salida detrás de nosotros y la pelotita ya no abandonará el campo de juego. Otras letras nos permitirán duplicar el ancho de nuestro bate -y facilitarán la tarea- mientras que alguna otra lo devolverá a su tamaño original.
Asimismo en algunos casos el cuadrado marcado nos permitirá disponer de dos bates -uno delante de otro- o quizás se volverá un objeto explosivo, que al ser tocado se destruirá a sí mismo y a todos los adyacentes, adelantando nuestro trabajo.
Aquí no se acaba todo. En cada pantalla nos encontraremos con rostros que flotan y desvían la trayectoria de la pelota; luces de colores que arrojan lluvia luminosa; culebras que realizan mil estropicios y muchas otras variantes.
Hay que tener, sin embargo, especial cuidado con dos de esos objetos extraños. El primero, una especie de pescado volador que sale disparado en nuestra dirección desde un OVNI. Si toca nuestro bate, lo paraliza y deberemos observar con impotencia cómo nuestra pelota se va por detrás sin que podamos hacer nada.
Los otros son unos antipáticos globulitos verdes,



jocosamente parecidos a los famosos "ensolves" que popularizó la publicidad de un jabón en polvo, y con aficiones parecidas. No harán otra cosa, ni más ni menos, que perseguir nuestra pelotita y, si logran alcanzarla, la devorarán con fruición y rapidez, para luego escupir sobre nuestro bate -con maldad sin igual- apenas un carozo.

RANKING DE SOFTWARE

Por votación de los lectores se elegirán los cinco mejores programas de juegos o utilitarios creados para la computadora Drean Commodore 64C.
Para participar se debe enviar el cupón (o fotocopia) a nuestra Redacción: Paraná 720, 5° Piso, Cap. Fed. (1017), personalmente o por correo.







1º- 1942
2º- GHOST'N GOBLINS
3º- COMMANDO
4º- ACE OF ACES
5º- INTERNATIONAL KARATE

PREMIO: 10 CASETES - 1 por ganador

Los ganadores deel concurso de ranking de software son: **Rodrigo Paz Lemos,** Ezpeleta. **Verónica Olmos,** Hurlingham. **Lorena Torres,** Quilmes. **Daniel Divito,** Escobar. **Marcelo Spiedo,** Ituzaingó. **Cristian Rojas,** Martínez. **Diego Longo,** Moreno. **Sergio de los Santos,** Baradero. **Martín P. Walser,** Capital. **María Cristina Cittiero,** Capital.



Entre los que envíen sus cupones se sortearán mensualmente:

Diez Software para Drean Commodore 64C

El franqueo para enviar premios es por contrareembolso a pagar en destino (o similar).

CUPON RANKING DE SOFTWARE DREAN COMMODORE

APELLIDO .... NOMBRE .... 1º ....
DIRECCION .... LOCALIDAD .... 2º ....
PROVINCIA .... C.P. .... T.E. .... 3º ....
OCUPACION .... EDAD .... 4º ....
COMPUTADORA .... 5º ....
QUE ES LO QUE MAS ME GUSTA DE LA REVISTA DEL USUARIO DE DREAN COMMODORE
....
QUE ES LO QUE MENOS ME GUSTA ....
....

C-16

Poseo una C-16 y me gustaría que indicaran dónde podría conseguir juegos y joysticks especialmente diseñados para C-16.

Javier Andrés Ernst
Santiago Marzol 768
6300 - Santa Rosa (La Pampa)



Podés ponerte en contacto con el Centro de Atención al Usuario. Publicamos además tu dirección para que otros usuarios, o ex-usuarios a los que les haya quedado material, se pongan en contacto con vos.

CONCURSO

Soy aficionado a la computación y me interesó mucho el concurso que cerrará en Noviembre. No quiero ofenderlos con la pregunta pero, ¿cómo hacen para estudiar todos los trabajos que llegan, pues pienso que es muy difícil que un jurado realice en poco tiempo la tarea de calificar los trabajos?

Gabriel Sberna
Capital Federal

No nos ofende la pregunta y te pasamos a aclarar la tarea del jurado, que no es uno solo y que ya comenzó a analizar los trabajos recibidos. Estos jurados se basan primeramente en criterios de selección según la originalidad de la idea, los efectos gráficos y sonoros, la presentación, la aplicación, la técnica de programación. Con estos criterios se definen los finalistas y de la reunión de todos los jurados y el análisis integral del programa sale el resultado.

Continuamos con esta sección para que los lectores planteen sus consultas y sugerencias. Para eso deben escribir a Revista para usuarios de Drean Commodore, Paraná 720, 5to. Piso, (1017) Cap.

INTERCAMBIO

Me agradaría contactarme con toda persona que tome la computación en serio (y no a simple nivel de video juego) para intercambiar programas utilitarios, ideas y trucos.
Soy maestro mayor de obras y me gustaría intercambiar programación técnica con colegas.
¿Dónde se "esconde" el Club de Usuarios Drean Commodore en la ciudad de Córdoba?.

Raúl Angel Lombo
Casilla de Correo 421
5152 - Villa Carlos Paz Córdoba

1. Trancribimos tu carta e invitamos a tus colegas a que se pongan en contacto con vos. En especial, recordemos que en el número 20 nos escribió Armando Butcovic, de Mendoza quien también es maestro mayor de obras.
2. Ante tu consulta, y la de otros usuarios del interior del país que tienen dificultad para conseguir asesoramiento y software, te recomendamos ponerte en contacto con el señor Pablo Suárez, del Centro de Atención al Usuario Drean Commodore de Buenos Aires, que se encarga de atender todas las consulta que provienen del interior del país.

SIMONS' BASIC

1. ¿En el SIMONS' BASIC los comandos para dibujar en alta resolución son iguales o parecidos a los del BASIC 7.0 de la C-128?
2. ¿El SIMONS' BASIC que se vende en disco es el mismo que se vende en cartucho?
3. Me gustaía que publicaran explicaciones exhaustivas de los juegos.

Juan Granillo
San Isidro

1. El SIMONS' BASIC fue creado y desarrollado por David Simons, de 16 años, para ampliar el BASIC estándar de la Commodore 64, la versión 2.0. De esta manera el usuario dispone de un total de 114 comandos. La C-128 tomó la mayoría de los nuevos comandos, entre ellos los gráficos, y agregó otros para hacer la versión 7.0.
2. Es el mismo tanto, en disco, casete o cartucho.
3. Tomamos en cuenta tu pedido.

AMIGA 500

En la revista número 20 se dice que la Commodore Amiga 500 tiene un precio sugerido de 649 dolares. ¿Dónde puedo conseguirla a ese precio? He consultado en un negocio que las importa y el precio de venta de ellos es de 1300 dólares.

Ernesto Corvi
Capital Federal

El precio que se menciona es el básico en los Estados Unidos, sin impuestos. A eso hay que agregar los impuestos de importación y así se llega al precio que mencionas. Para más información sobre esta máquina te recomendamos que te comuniques con el Centro de Atención al Usuario Drean Commodore.

FOTO DE TAPA:
HANDBALL MARADONA



La foto de tapa corresponde a la pantalla de presentación del software **Handball Maradona** distribuido por Papillón y comentado en nuestra sección Revisión de Software del número anterior.



SEPTIEMBRE 1987

EL
COMPACT DISK INFORMATICO

MAS
SORTEOS Y CONCURSOS

SPECTRUM VS. TS-2068
HARD

C-64/128
PROCESADORES DE TEXTO

PC
NUEVOS MODELOS

AÑO 3 Nº 30 ₳ 5 REP. ARGENTINA